Atacama
Large
Millimeter
Array

Interface Control Document

between:

AEM Antenna

and:

ALMA Computing, Control Software

ALMA-33.00.00.00-70.35.20.00-A-ICD

Version: A dated 05.08.2011
Status: Draft

Prepared By:	Organization	Date
E. Allaert and P. Martinez	ESO	2011-08-05
IPT Leader Approvals:	**Organization**	**Date**
Stefano Stanghellini Gianni Raffi	ESO ESO	2011-08-05 2011-08-05
System Engineering Approvals:	**Organization**	**Date**
Christoph Haupt Javier Marti-Canales	ESO JAO	2011-08-05 2011-08-05
Configuration Control Board Approval:	**Organization**	**Date**
Javier Marti-Canales	ALMA Configuration Control Board Secretary, signing for the Control Board	2011-08-05
JAO Director Release Authorization:	**Organization**	**Date**
Thijs de-Graauw	Joint ALMA Office Project Director	2011-08-05

Change Record

Version	Date	Affected Section(s)	Change Request #	Reason/Initiation/Remarks
A	2003-10-10	All		Initial release
B	2003-12-14	Applicable documents table and header logo	None	S. Oliver added in new alma logo in headers and changed all applicable documents to reference documents
C	2005-11-30	Sections 2, 3.1, 4.1, 4.2, 4.3, 4.4, 4.5, 5.1.1, 5.1.3.1, 5.1.3.2, 5.1.3.3, 5.1.3.4, 5.1.3.7, 5.2, 5.3, 5.5, 5.6 and 6 5.1.3.5, 5.1.3.6 and 5.6		Clarifications and minor corrections New sections
D	2006-12-08	Sections 5.1.3.1 and 5.1.3.2 5.1.3.3	ALMA-34.00.00.00-034-B-CRE	Additional subreflector mechanism tilt capability
E	2007-09-06	Sections 2.1 4.2 4.2.1 4.5 5.1.3.1 5.1.3.2 5.1.3.3 5.1.3.4 5.1.3.7 5.1.3.8 5.1.4.1		RD[01] updated to revision B Added Cabling Plan Text regarding Utility module type updated Changed from contact closed to contact open for alarms. Update 5 Stow monitor and 7 Over temperature alarm. Removed 9 48 VDC. Defined type of Utility module "power strip" removed Added ACU_TRK_MODE_RSP, GET_METR_DELTAS_TEMP, GET_METR_DELTAPATH, SELFTEST_ERR, Modified GET_METR_EQUIP_STATUS, Modified GET_METR_DISPL_N. Added ACU_TRK_MODE ACU_MODE_RSP added SELFTEST mode, Added ACU_TRK_MODE_RSP GET_ACU_ERROR, added error codes GET_METR_EQUIP_STATUS, changed to 4 bytes, added bit for blanking GET_METR_DISPL_N, changed to 4 bytes GET_METR_TEMPS_N, added sensor fault Added GET_METR_DELTAS_TEMP Added GET_METR_DELTAPATH GET_ANTENNA_TEMPS and GET_ANTENNA_TEMPS, added sensor fault values

Version	Date	Affected Section(s)	Change Request #	Reason/Initiation/Remarks
		5.6 5.7		Changed data for SELFTEST_RSP Added SELFTEST_ERR Added ACU_TRK_MODE Updated description for SELFTEST_CMD New section (Trajectory Commands) New section (OTP Mapping) New section (Tracking sub mode) Rewritten New section (Circular buffer)
AEM-F	2010.04.22			▪ GET_SERIAL_NUMBER added ▪ INIT_AZ_ENC_ABS_POS added ▪ INIT_EL_ENC_ABS_POS added ▪ GET_METR_DISPL_N eliminated (not used by AEM) ▪ SET_AIR_CONDITIONING added ▪ GET_PT_MODEL_COEFF_N confirmed ▪ SET_PT_MODEL_COEFF_N confirmed ▪ Hexapod hardware limits eliminated ▪ SELF TEST GET/SET updated ▪ GET_METR_COEFF_N added ▪ SET_METR_COEFF_N added
AEM-G	2011.03.15	5.1.3.1-5.1.3.3 5.1.3.2-5.1.3.4 5.1.4 5.4 5.6 6		Monitor points general update Control points general update Emergency Stow eliminated Reference to S/W Mainten. Manual added. Self Test References added.
AEM-G	2011.03.22	 5.1.3.1-5.1.3.2		Minor refurbishments on the basis of ESO comments. Page numbering update on tables 5 and 6.
AEM-G	2011.04.06	5.1.3.1-5.1.3.2 5.1.3.3 - GET_AZ_STATUS - GET_EL_STATUS 5.1.3.4 - RESET_ACU_CMD_1		Automatic page numbering update inserted on tables 5 and 6. Minor refurbishments on the basis of ACRV#01 actions.
AEM-A New number	2011.04.11	1.1		Document renumbered as requested by ESO EL Stow **Pins** command returned **Pin**
AEM-A New number	2011.07.22	5.1.3.3		Added GET_AC_TEMP Added DUMP_ERROR_LOG *GET_SYSTEM_STATUS updated* *SELFTEST_CMD_1 updated*

Version	Date	Affected Section(s)	Change Request #	Reason/Initiation/Remarks
AEM-A New number	2011.08.05	All		Final refurbishments after ESO comments for official release towards ALMA

Table of Contents

1 Description

1.1 Purpose

The purpose of this document is to define the interface between the AEM antenna and specifically its control unit and ALMA's monitor and control (M&C) system. This ICD provides the interface definitions for the minimum control functionality which is identified at present for the control of the antenna. Additional functionality will have to be added by the Antenna Contractor in agreement with ALMA to take into account their design, and in particular aspects linked to monitoring and safety.

2 Related Documents and Drawings

2.1 Reference Documents

[RD01] "ALMA Monitor and Control Bus Interface Specification", ALMA-70.35.10.03-001-B-SPE, Version B

[RD02] "CAN System Engineering", Wolfhard Lawrenz, Springer-Verlag, 1997 (Sections 1 & 2)

[RD03] "Coordinate System Specification", ALMA-80.05.00.00-009-B-SPE

[RD04] "ALMA Cabling Installation Plan for AEM Antenna Contractor", ANTD-80.04.00.00-019-B-PLA

[RD05] "Software Maintenance Manual Antenna Control System", ANTD-3335030-3-027-MAN

[RD06] "Software User Manual Antenna Control System", ANTD-3335030-3-026-MAN

[RD07] Antenna Control Unit Design Report ANTD-3335030-3-001-REP

2.2 Standards

[RD08] ANSI/IEEE Std 1014-1987. "IEEE Standard for a Versatile Backplane Bus: VME bus."

[RD09] ISO/IEC 8802-3: 1992 "Information Processing Systems - Local Area Networks - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD). Access methods and Physical Layer Specifications." Ethernet standard.

[RD10] ISO 11898:1993 Road vehicles - Interchange of digital information - Controller area network (CAN) for high-speed communication

[RD11] EIA Standard RS-485. “Standard for electrical characteristics of generators and receivers for use in balanced digital multipoint systems”. Electronic Industries Association, 1983.

3 Abbreviations and Acronyms

3.1 Glossary

ABM	*Antenna Bus Master*
Access Mode	*Current mode of accessing the ACU. When in Local access mode, the ACU may only be commanded by a local handset; all commands via the CAN M&C interface are ignored except for monitor requests. When in Remote access mode, M&C commands are accepted. The ACU may not be switched from Local to Remote access mode over the CAN M&C interface.*
ACK	*Acknowledge. In a CAN transmission, this is a bit in a transmitted frames which is set by a successful receiver of the frame*
ACU	*Antenna Control Unit*
ALMA	*Atacama Large Millimeter Array*
AUI	*A type of Ethernet connector*
AZ	*Azimuth. Defined as zero to North.*
BE	*Back-End*
Bore sight	*The actual orientation of the axis of symmetry of the main reflector with respect to established local coordinates (zenith direction and nominal azimuth zero).*
CAN	*Controller Area Network*
CAN ID	*CAN message identification. A 29 bit identifier transmitted at the start of a CAN frame which also determines the frame's priority*
CDR	*Critical Design Review*
EL	*Elevation*
FE	*Front End*
ICD	*Interface Control Document*
ISO	*International Standards Organisation*
LAN	*Local Area Network*
M&C	*Monitor and Control*
Operational Mode	*The ACU state determining the availability of axis drive motors, and brakes. Also defines how the position commands are interpreted.*
PCU	*Portable (handheld) Control Unit*
RTR	*Remote Transmission Request. A type of CAN frame requesting transmission of a particular frame*
TBD	*To Be Determined*
Turns	*One turn of an antenna axis, or 360°*
UPS	*Un-interrupted Power Supply*
VME	*VERSAbus Module European*
VoIP	*Voice over IP*
WVR	*Water Vapor Radiometer*

4 Electronic Interface

4.1 Computer Hardware and Software

All embedded microprocessor systems shall be based on VME backplanes. All software shall be written for RTAI real-time Linux operating system. The exact RTAI/Linux version for the delivery of the software will be defined by ALMA, having discussed with the contractor, 6 months before provisional acceptance of the software. The source code shall be available to ALMA together with compiler and development tools. Additionally procedures for updating the ACU software remotely shall be provided.
The contractor is responsible for obtaining licenses needed for RTAI or other software used.
All application programming for processors in the control unit shall be written in C or C++. Executable code shall be stored in non-volatile electronic memory, avoiding mechanically driven peripherals such as disk drives.
All microprocessor systems shall have Ethernet interfaces for debugging and testing. All computers on the antenna shall be fully functional in the absence of any external connection to the Ethernet network.

4.2 Location

The Contractor shall locate the CAN bus interface connector (see Section 4.3) on his equipment in such a way that a cable from there to the center of the receiver room will not exceed 25 m in length, including, if necessary, the traversal of any cable wraps around the antenna motion axes. The absolute maximum length is 35m.

The contractor shall provide space for an Ethernet switch (provided and mounted by ALMA) in a temperature controlled and easily accessible area, preferably the azimuth platform or pedestal room, but not in the receiver cabin. The switch shall be connected to a separate UPS power used only for the switch and the utility module, see below. The UPS power shall be provided by the contractor.

A remotely operated utility module (see below) with Ethernet interface shall be mounted close to the Ethernet switch. The contractor shall procure, mount and test the utility module. Software for testing the utility module to validate its functionality shall also be provided.

The Antenna Bus Master (ABM) will be provided and mounted by ALMA in the BE rack in the receiver cabin.
A CAN bus shall be routed by the contractor from the ACU to the ABM in the receiver cabin. Another CAN bus shall be routed by the contractor from the ABM to the apex (the position where the subreflector is mounted). The contractor shall also route a CAN cable from the ABM to the optical telescope.

Place shall also be reserved for two VoIP telephones in a temperature controlled area, one in the receiver cabin and the other in the pedestal room.

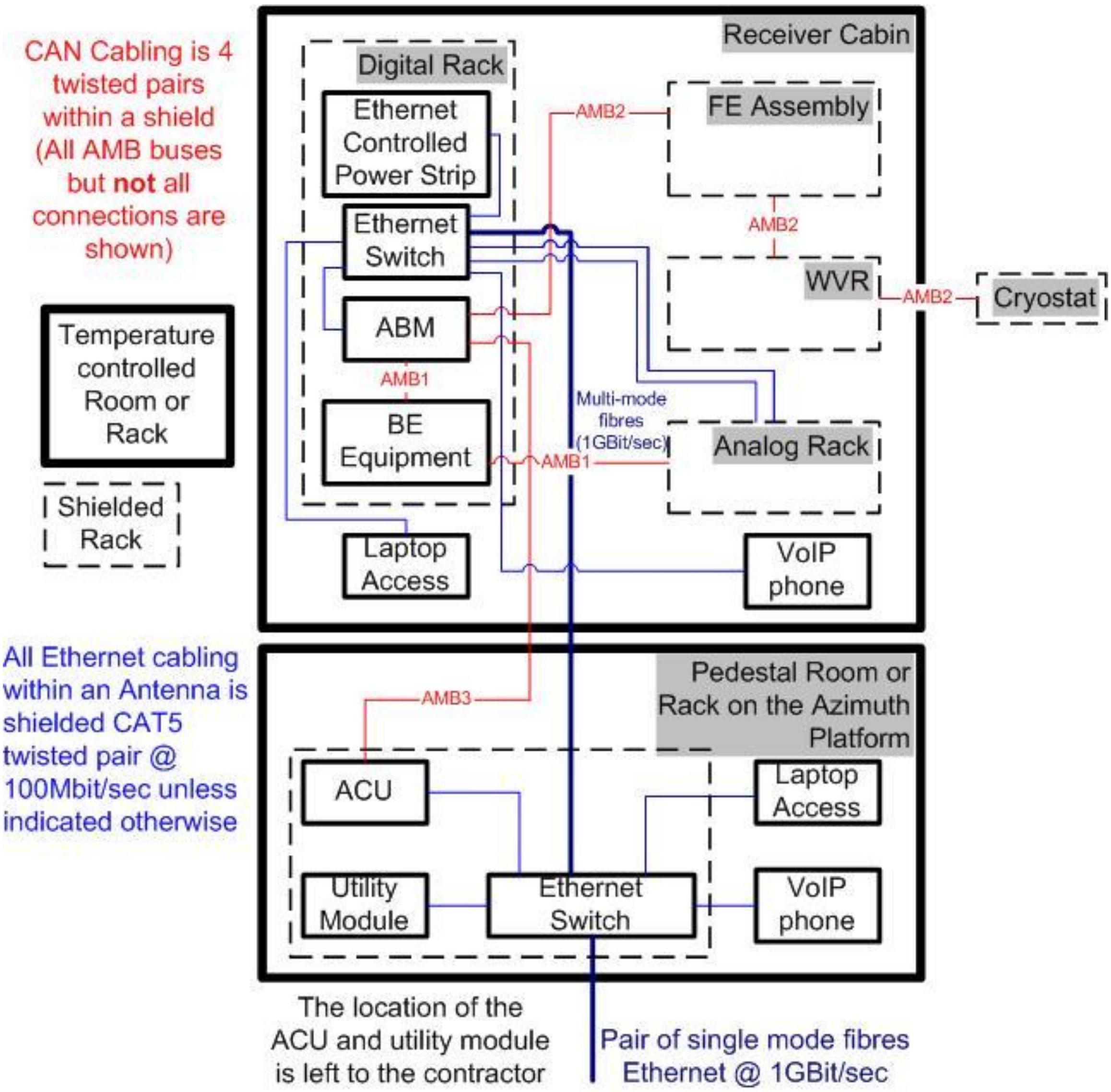

Figure 2: Cable Routing. For actual cabling, see [RD04]

4.2.1 Utility Module

The utility module is a device to remotely through Ethernet monitor and control digital inputs and outputs. The inputs/outputs shall be galvanically isolated.

The utility module shall receive the following alarm conditions and relay the status information to the Ethernet via an IP address:

1. Fire Alarm (Contact closure in normal state directly to utility module from battery-

operated fire sensor. Contact open in case of alarm.)
2. Emergency-stop set (Contact closure in normal state directly to utility module from emergency stop chain. Contact open if any of the emergency-stop buttons are pushed in.)
3. AC power off (Contact closure in normal state directly to utility module from AC power unit. Contact open in the event AC power is off.)
4. UPS power off (Contact closure in normal state directly to utility module from UPS power unit. Contact open in the event UPS power is off.)
5. Stow monitor (Contact closure in normal state directly to utility module from stow pins. Contact open in the event any stow pin is in.)
6. Over temperature alarm (Contact closure in normal state directly to utility module from receiver cabin and motor temperature sensors. Contact open in the event of over temperature in receiver cabin or motors.)
7. HVAC alarm (Contact closure in normal state directly to utility module from HVAC unit. Contact open in the event HVAC unit is powered down.)
8. Zenith pointing (Contact closure in normal state directly to utility module from antenna zenith pointing sensor. Contact open in the event antenna is at zenith pointing.)
9. Spare.
10. Feed shutter. Contact closure in normal state directly to utility module from feed shutter. Contact open in the event of feed shutter being open when elevation angle is above 89.0 degrees.
11. Spare (up to 12 inputs are possible with example device)

Note that status information must pass directly to the utility module whether or not it also passes to a control computer.

The utility module shall receive the following commands from the Ethernet IP and output a control bit to the following functions:

1. Remote set emergency stop. (Contact closure in normal state directly to emergency stop activator in ACU. Contact open to activate emergency stop. The emergency stop activation shall be independent of the ACU operation. This function MAY NOT disengage any locally set emergency stop.) Reset of this emergency stop may not change the state of the antenna.
2. Spare (Two outputs are possible with example device).

In addition, for safety reasons, the input signal logic shall be ‘contact closure on normal state, e.g. contact open on alarm’.

The utility module must have an AC power source separate from the Antenna AC power or the main UPS power. Power must be provided for current loops for each contact closure.

Type of utility module device: ADAM-6051, Data Acquisition Module, available from B&B Electronics, www.bb-elec.com.

4.3 Monitor and Control Interface

The serial bus interfaces between ALMA's system and the Contractor's ACU and shall be a CAN bus as described in [RD01]. The connector type shall be a 9-pin D shell connector with pin allocation as given in **Section 2.1.2** of [RD01]. The diagram is reproduced here and represents the bus stub connector. The ACU shall provide one female and one male connector for connection to the CAN bus making it possible to daisy chain the CAN bus. The contractor shall supply external terminator for the CAN bus to be attached to the output CAN connector. The CAN connectors shall be easily accessible when the ACU is mounted.

Note that the reset signal (pins 1 and 6) and the time signal (pins 4 and 8) are non-standard but are required. Pin 9 is defined in the CAN standard for use in supplying power to bus devices. It will not be used for this purpose within ALMA.

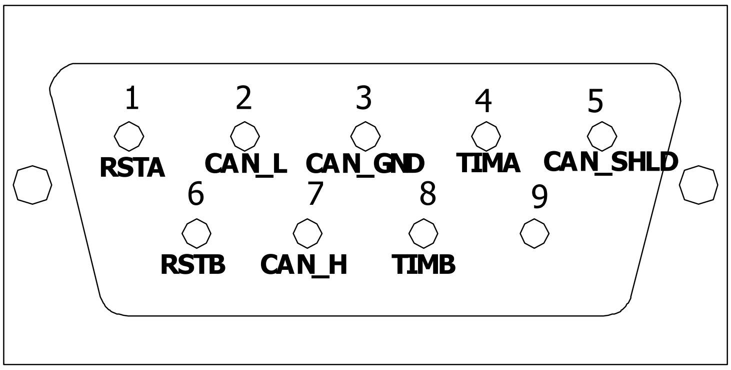

Figure 1: CAN D-connector pin allocations. A male connector is shown, viewed from the pin side

Table 1: CAN D connector pin definitions

Pin	Signal	Description
1	RSTA	Global Slave Node Reset, line A
2	CAN_L	CAN_L bus line (dominant low)
3	CAN_GND	CAN Ground
4	TIMA	Timing Signal, line A
5	CAN_SHLD	CAN Bus Shield
6	RSTB	Global Slave Node Reset, line B
7	CAN_H	CAN_H bus line (dominant high)
8	TIMB	Timing Signal, line B
9	-	Reserved

4.4 Timing Interface

In addition to the serial bus, the ACU will receive a precise timing reference signal on the same CAN D-connector as shown in Figure 1 and Table 1 This will be a periodic pulse, supplied by differential signaling conforming to RS485. The pulse period shall be 48

milliseconds and width of 5 to 10 milliseconds, with a nominal value of 6 milliseconds. The leading edge of each pulse marks a timing event.

The source will contain a RS485 transmitter, which will drive the bus to a quiescent state of logic 0 (FALSE), and will drive it to a logic 1 (TRUE) periodically with a duty cycle of 12.5 %. The period is specified to be 48.0 ms. Use of the signal at other nodes is optional, but each user node shall have a RS485 receiver that is designed so that an open circuit or short circuit is interpreted as logic 0. The leading edge (0 to 1 transition) of the signal will be accurately synchronized to ALMA array time (with a maximum error to be specified elsewhere), but the timing of the falling edge (1 to 0) is not specified.

Further details are available in [RD02]. Note that the "TIMA" and "TIMB" signals shown in Figure 1 denote the RS422 A and B lines as defined in [RD08].

4.5 Ethernet Interface

The Ethernet interface, to be provided for debugging, maintenance and software updates, shall conform to [RD06]. Ethernet cables shall be routed in such a way that the bending radius allows for later installation of fiber optic cables

Ethernet cabling shall be provided for the ACU, utility module and the VoIP telephones. Additionally there shall be three Ethernet connections for laptops, one in the receiver cabin, one on azimuth platform and one in the pedestal room. All Ethernet cabling within an antenna shall be shielded CAT5 twisted pair for 100 Mbit/sec unless indicated otherwise. The Ethernet switch shall be connected to another Ethernet switch in the BE rack in the receiver cabin. This cable shall be multi-mode fiber specified for 1 GBit/sec Ethernet.

5 Software Control Function Interface

5.1 Monitor and Control Software Interface

5.1.1 General

The CAN bus in use for monitor and control by ALMA consists of the CAN 2.0B variant and a non-standard higher level protocol defined in [RD02]. CAN 2.0B specifies the extended, or 29 bit, address range for the CAN frame [RD02]. The implications of the higher level protocol will be discussed further in this section. The baud rate of the CAN bus is 1 Mbits/sec.
Unless explicitly stated otherwise, all M&C values (integer, fixed or floating point) shall be transmitted in network byte order, or big endian fashion. Where specific bits in a byte are referenced in the following sections, bit 0 is the least significant bit.
In accordance with [RD02], the Contractor's ACU CAN shall have a unique 64 bit serial number and the CAN node address 0. The node address defines a range of CAN addresses within which all of the ACU specific CAN traffic will fall. As the ACU is a special node, ALMA defines the serial number and node address as follows:

Table 2: ACU Bus Constants, to be updated for production antennae

	Value (hexadecimal)
ACU Serial Number	*Unique 64 bit serial number for each ACU*
ACU Node Address	*0*

In the following sections, the prefix “0x” denotes that the number is hexadecimal.
This implies that the ACU will respond to the Identify Node broadcast message on CAN ID 0x00000000 as defined in the following table:

Table 3: ACU Bus Identify Response, to be updated for production antennae

ACU receives CAN ID (hex)	*ACU transmits CAN ID (hex)*	*With data bytes (hex)*
00 00 00 00	*00 04 00 00*	*Unique 64 bit ACU serial number*

Note that these transmissions must begin within 1 millisecond of receiving the Identify Node broadcast message.

The range of CAN IDs to which the ACU responds for M&C data will be restricted to
0x00 04 00 00 to 0x00 07 FF FF.

As defined in [RD01], each defined CAN ID represents a single monitor or control point. Control points require no explicit acknowledgement beyond the implicit CAN ACK bit. Monitor data is requested by a zero-length frame (not an RTR frame) and the ACU must respond with the appropriate monitor data within 150 microseconds. If the values to be returned are not time critical, they may be returned from a local cache.

All commands (control points designated by a CMD suffix) must result in success, an error or a timeout. These error conditions shall be stored on a stack, which may be polled with the GET_ACU_ERROR monitor point. This monitor request returns an error from the stack until none are left. All errors include a 1 byte code identifying the error condition or timeout and a 4 byte identification of the command which caused the error or timeout. The stack shall be big enough to contain at least 32 errors. If the stack is full new errors shall be discarded and a specific error shall indicate the stack overflow.
If unexpected commands or commands or monitor points with not defined CAN id are received, an error shall be put on the error stack.

The Contractor's interface shall respond correctly if up to 50 messages per 48 millisecond timing period are addressed to it. An overall message rate on the CAN bus (including messages addressed to other nodes) that uses the full 1 Mbits/sec raw data rate shall not cause any errors in the Contractor's interface.
The following tables summarize all M&C points for the ACU, with their CAN ID allocations, data size and typical access rates. Each M&C point is then described in more detail after section 5.1.2, which details the data types used in the detailed descriptions.

Please note that this list of monitor and control points is expandable during the antenna design phase and will be frozen at the antenna CDR.

5.1.2 Data Types

The following table indicates the data types used within CAN messages.

Table 4: CAN Data Types

Data Type	***Description***
Bit	*A single bit within an ubyte. Unused bits within ubytes are padded to fill the byte.*
ubyte	*An unsigned byte, usually used for bit fields.*
int8	*A signed 8 bit integer value.*
uint8	*An unsigned 8 bit integer value.*
int16	*A signed 16 bit integer value.*
uint16	*An unsigned 16 bit integer value.*
int32	*A signed 32 bit integer value.*
uint32	*An unsigned 32 bit integer value.*
float	*Single precision 32 bit IEEE floating point value.*
double	*Double precision 64 bit IEEE floating point value.*
string	*A string of single byte characters. Length is given by the DLC field in the CAN frame and the string is* ***not*** *null terminated.*

When multiple types are used in a single CAN message payload, there is no padding between values in a message.
All values appear in the CAN message payload in network byte order, or most significant byte first. Within a byte, bit 0 is the least significant bit.

5.1.3 ACU M&C Points

The following tables summarize all M&C points for the ACU, with their CAN ID allocations, data size and typical access rates. Each M&C point is then described in more detail.

5.1.3.1 Summary of ACU Monitor Points

Monitor data shall be polled by the ALMA bus master according to the protocol specified in [RD01]. A monitor request consists of a transmission of the appropriate CAN message with zero bytes of data. The ACU shall respond within 150 microseconds by transmitting the requested data in a message with the same CAN identification.

The "_RSP" suffix denotes a response message for which a corresponding "_CMD" control point exists. Most, but not all "GET_" monitor points have a corresponding "SET_" control point.

Table 5: Summary of Monitor Points

Name	*CAN ID (hex)**	*Data Size (bytes)*	Typical Interval *(secs)*	Page Number
GET_SERIAL_NUMBER	*00 04 00 00*	*8*	*Rare (at ABM startup)*	19
ACU_MODE_RSP	*00 04 00 22*	*2*	*5*	19
ACU_TRK_MODE_RSP	*00 04 00 20*	*1*	*5*	19
AZ_POSN_RSP	*00 04 00 12*	*8*	*0.048*	20
EL_POSN_RSP	*00 04 00 02*	*8*	*0.048*	20
GET_ACU_ERROR	*00 04 00 2F*	*0 or 5*	*0.048*	20
GET_AZ_TRAJ_CMD	*00 04 00 13*	*8*	*Rare*	21
GET_AZ_BRAKE	*00 04 00 14*	*1*	*5*	21
GET_AZ_ENC	*00 04 00 17*	*4*	*0.048*	21
GET_AZ_MOTOR_CURRENTS	*00 04 00 19*	*4*	*5*	22
GET_AZ_MOTOR_TEMPS	*00 04 00 1A*	*4*	*5*	22
GET_AZ_MOTOR_TORQUE	*00 04 00 15*	*4*	*5*	22
GET_AZ_SERVO_COEFF_N	*00 04 30 20 – 00 04 30 2F*	*8*	*Rare*	22
GET_AZ_STATUS	*00 04 00 1B*	*8*	*5*	23
GET_AZ_ENCODER_OFFSET	*00 04 00 1C*	*4*	*Rare*	24
GET_CAN_ERROR	*00 07 00 01*	*4*	*(debug)*	24
GET_EL_TRAJ_CMD	*00 04 00 03*	*8*	*Rare*	25
GET_EL_BRAKE	*00 04 00 04*	*1*	*5*	25
GET_EL_ENC	*00 04 00 07*	*4*	*0.048*	25
GET_EL_MOTOR_CURRENTS	*00 04 00 09*	*4*	*5*	26
GET_EL_MOTOR_TEMPS	*00 04 00 0A*	*4*	*5*	26
GET_EL_MOTOR_TORQUE	*00 04 00 05*	*4*	*5*	26
GET_EL_SERVO_COEFF_N	*00 04 30 10 – 00 04 30 1F*	*8*	*Rare*	26
GET_EL_STATUS	*00 04 00 0B*	*8*	*5*	27
GET_EL_ENCODER_OFFSET	*00 04 00 0C*	*4*	*Rare*	28
GET_SYSTEM_ID	*00 07 00 04*	*3*	*Rare*	28
GET_IDLE_STOW_TIME	*00 04 00 25*	*2*	*Rare*	29
GET_IP_ADDRESS	*00 04 00 2D*	*8*	*Rare*	29
GET_IP_GATEWAY	*00 04 00 38*	*4*	*Rare*	29
GET_NUM_TRANS	*00 07 00 02*	*4*	*(debug)*	29
GET_SYSTEM_STATUS	*00 04 00 23*	*7*	*5*	29
GET_PT_MODEL_COEFF_N	*00 04 30 40 - 00 04 30 5F*	*8*	*Rare*	31
GET_SHUTTER	*00 04 00 2E*	*1*	*5*	31
GET_STOW_PIN	*00 04 00 24*	*2*	*5*	32

Name	*CAN ID (hex)**	*Data Size (bytes)*	Typical Interval *(secs)*	Page Number
GET_STOW_PIN_1	*00 04 00 21*	*4*	*5*	*32*
GET_SUBREF_ABS_POSN	*00 04 00 26*	*6*	*5*	*33*
GET_SUBREF_DELTA_POSN	*00 04 00 27*	*6*	*5*	*33*
GET_SUBREF_LIMITS	*00 04 00 28*	*3*	*5*	*34*
GET_SUBREF_ROTATION	*00 04 00 2A*	*6*	*Rare*	*35*
GET_SUBREF_STATUS	*00 04 00 29*	*4*	*5*	*35*
GET_METR_MODE	*00 04 00 31*	*4*	*Rare*	*36*
GET_METR_EQUIP_STATUS	*00 04 00 32*	*4*	*5*	*36*
GET_METR_COEFF_N	*00 04 00 50 – 00 04 00 51*	*8*	*Rare*	*37*
GET_METR_TEMPS_N	*00 04 40 N*	*8*	*5*	*37*
GET_METR_TILT_N	*00 04 50 N*	*8*	*5*	*37*
GET_METR_DELTAS	*00 04 00 34*	*8*	*0.048*	*38*
GET_METR_DELTAS_TEMP	*00 04 00 33*	*8*	*0.048*	*38*
GET_METR_DELTAPATH	*00 04 00 53*	*4*	*0.048*	*38*
GET_POWER_STATUS	*00 04 00 30*	*2*	*5*	*38*
GET_AC_STATUS	*00 04 00 2C*	*8*	*5*	*39*
GET_AC_TEMP	*00 04 00 2B*	*4*	*Rare*	*41*
GET_UPS_OUTPUT_VOLTS	*00 04 00 35*	*6*	*5*	*41*
GET_UPS_OUTPUT_CURRENT	*00 04 00 36*	*6*	*5*	*41*
GET_ANTENNA_TEMPS	*00 04 00 37*	*4*	*5*	*41*
GET_SW_REV_LEVEL	*00 07 00 00*	*3*	*(debug)*	*42*
SELFTEST_RSP	*00 04 00 40*	*5*	*Rare*	*42*
SELFTEST_ERR	*00 04 00 41*	*6*	*Rare*	*43*
SELFTEST_ERR_1	*00 04 00 42*	*8*	*Rare*	*43*

5.1.3.2 Summary of Control Points

Control data shall be transmitted by the ALMA bus master according to the protocol specified in [RD01]. A control transaction consists of a transmission of the appropriate CAN message with data, if appropriate. The ACU shall acknowledge receipt of the control message by setting the acknowledge bits in the trailer of the CAN transmission. No further response is required.

Note that command failures and error conditions are polled in the monitor message GET_ACU_ERROR.
The "_CMD" suffix denotes a command message which should result in the ACU adding error or timeout message on the error stack should the command fail. All of the "SET_" control points have a corresponding "GET_" monitor point.

Command input parameters shall be checked for valid range and violations shall be reported on the error stack. Commands shall be rejected if not all conditions for their execution are fulfilled and an error shall be put on the error stack.

Table 6: Summary of Control Points

Name	*CAN ID (hex)**	*Data Size (bytes)*	*Typical Interval (secs)*	*Page Number*
ACU_MODE_CMD	*00 04 10 22*	*1*	*Rare*	*44*
ACU_TRK_MODE_CMD	*00 04 10 20*	*1*	*Rare*	*44*
AZ_TRAJ_CMD	*00 04 10 12*	*8*	*0.048*	*44*
EL_TRAJ_CMD	*00 04 10 02*	*8*	*0.048*	*45*
CLEAR_FAULT_CMD	*00 04 10 21*	*1*	*Rare*	*45*
RESET_ACU_CMD	*00 04 10 2F*	*1*	*Rare*	*45*
RESET_ACU_CMD_1	*00 04 10 32*	*1*	*Rare*	*46*
SET_AZ_BRAKE	*00 04 10 14*	*1*	*Rare*	*46*
SET_AZ_SERVO_COEFF_N	*00 04 20 20 – 00 04 20 2F*	*8*	*Rare*	*46*
SET_AZ_SERVO_DEFAULT	*00 04 10 17*	*1*	*Rare*	*47*
INIT_AZ_ENC_ABS_POS	*00 04 10 18*	1	*Rare*	*47*
SET_EL_BRAKE	*00 04 10 04*	*1*	*Rare*	*47*
SET_EL_SERVO_COEFF_N	*00 04 20 10 – 00 04 20 1F*	*8*	*Rare*	*47*
SET_EL_SERVO_DEFAULT	*00 04 10 07*	*1*	*Rare*	*48*
INIT_EL_ENC_ABS_POS	*00 04 10 08*	*1*	*Rare*	*48*
SET_IDLE_STOW_TIME	*00 04 10 25*	*2*	*Rare*	*48*
SET_IP_ADDRESS	*00 04 10 24*	*8*	*Rare*	*48*
SET_IP_GATEWAY	*00 04 10 38*	*4*	*Rare*	*48*
SET_PT_MODEL_COEFF_N	*00 04 20 40 – 00 04 20 5F*	*8*	*Rare*	*49*
SET_STOW_PIN	*00 04 10 2D*	*2*	*5*	*49*
SET_SUBREF_ABS_POSN	*00 04 10 29*	*6*	*0.5*	*49*
SET_SUBREF_DELTA_POSN	*00 04 10 2A*	*6*	*0.5*	*50*
SUBREF_DELTA_ZERO_CMD	*00 04 10 2B*	*1*	*0.5*	*50*
SET_SUBREF_ROTATION	*00 04 10 28*	*6*	*Rare*	*50*
SET_METR_MODE	*00 04 10 26*	*4*	*Rare*	*51*
SET_METR_COEFF_N	*00 04 10 50 – 00 04 10 51*	*8*	*Rare*	*51*
SET_SHUTTER	*00 04 10 2E*	*1*	*Rare*	*51*
SELFTEST_CMD	*00 04 10 30*	*1*	*Rare*	*52*
SELFTEST_CMD_1	*00 04 10 31*	*1*	*Rare*	*52*
SET_AIR_CONDITIONING	*00 04 10 27*	*6*	*Rare*	*53*
DUMP_ERROR_LOG	*00 04 10 70*	*1*	*Rare*	*53*

The letter “N” is a hexadecimal digit in the range [0, F].

5.1.3.3 Monitor Points in Detail

Name *GET_SERIAL_NUMBER*
CAN ID *00 04 00 00*
Description *Return the device specific serial number*
Typical Interval *5s*
Data *uint8[8]*
The returned number is same as the serial number returned when this device responds to an identify broadcast

Name *ACU_MODE_RSP*
CAN ID *00 04 00 22*
Description *Current* operational *and access mode information for ACU*
Typical Interval *5 s*
Data *2 bytes:*
byte 0: Axis Modes: ubyte
bits 0-3: Azimuth Mode
bits 4-7: Elevation Mode
Axis Mode values:
0x0 SHUTDOWN
0x1 STANDBY
0x2 ENCODER
0x3 AUTONOMOUS
0x4 SURVIVAL STOW
0x5 MAINTENANCE STOW
0x6 VELOCITY
0x7 SELFTEST
byte 1 Access Mode: ubyte
0x01 LOCAL
0x02 REMOTE

Name *ACU_TRK_MODE_RSP*
CAN ID *00 04 00 20*
Description *Current tracking mode information for ACU*
Typical Interval *5 s*
Data *1 byte: Axis Tracking Modes: ubyte*
Axis Tracking Mode values:
0x0 CONTINUOUS SIDEREAL TRACKING
0x1 SLEWING
0x2 FAST SWITCHING
0x3 ON THE FLY TOTAL POWER MAPPING
0x4 ON THE FLY TOTAL INTERFEROMETRIC MOSAICKING

Name	*AZ_POSN_RSP*
CAN ID	*00 04 00 12*
Description	*Position of azimuth axis in turns at the last 20.83Hz pulse and 24ms before. Note that the interpretation of the value depends on the current active mode. In ENCODER mode, the position values are uncorrected; in AUTONOMOUS mode the values have been corrected by pointing model and metrology.*
Typical Interval	*48 ms*
Data	*8 bytes:* *bytes 0-3: azimuth position at the last 20.83 Hz timing pulse* *bytes 4-7: azimuth position 24 ms before the last timing pulse* *Data format: signed, two's complement, fixed point binary number representing angle from –1 turn to* $+(1–2^{-31})$ *turn.*

Name	*EL_POSN_RSP*
CAN ID	*00 04 00 02*
Description	*Position of elevation axis in turns at the last 20.83Hz pulse and 24ms before. Note that the interpretation of the value depends on the current active mode. In ENCODER mode, the position values are uncorrected; in AUTONOMOUS mode the values have been corrected by pointing model and metrology.*
Typical Interval	*48 ms*
Data	*8 bytes:* *bytes 0-3: elevation position at the last 20.83 Hz timing pulse* *bytes 4-7: elevation position 24 ms before the last timing pulse.* *Data format: signed, two's complement, fixed point binary number representing angle from –1 turn to* $+(1–2^{-31})$ *turn.*

Name	*GET_ACU_ERROR*
CAN ID	*00 04 00 2F*
Description	*ACU error conditions. This monitor point returns an error stack which includes an error code and an identification of the command causing the error.*
Typical Interval	*48 ms*
Data	*5 Bytes (0 bytes in case of no error):* *Error condition indicated as follows:* *byte 0 (ubyte): Error code:* *0x00: No error;* *0x01: Timeout* *0x02: Invalid mode change requested* *0x03: Requested position out of range* *0x04: Requested velocity out of range* *0x05: ACU in Local Access Mode* *0x06: Invalid brake command requested*

Name *GET_ACU_ERROR*
CAN ID *00 04 00 2F*
Description *ACU error conditions. This monitor point returns an error stack which includes an error code and an identification of the command causing the error.*
Typical Interval *48 ms*

0x10:Illegal command or monitor request (undefined CAN ID)
0x11:Unexpected command or monitor request
(if a command arrives when it is not allowed)
0x12: Parameter out of range
0x13: Invalid data length of command
0x14: Trajectory command delayed
(if the trajectory command for TE i+2 arrives after TE i + 24 ms)
0x15: Trajectory command duplicate error
(if two trajectory commands arrive within the same TE)
0x16: Error stack overflow
bytes 1-4 (uint32): Relative address of CAN message triggering error condition

Name *GET_AZ_TRAJ_CMD*
CAN ID *00 04 00 13*
Description *Position in turns and velocity in turns/sec set with the last AZ_TRAJ_CMD.*
Typical Interval *Rare*
Data *8 bytes.*
Bytes 0-3: Fixed point number as described in AZ_POSN_RSP representing turns.
Bytes 4-7: Fixed point number representing "velocity" in turns/sec.
Returns zero values if no AZ_TRAJ_CMD has been given.

Name *GET_AZ_BRAKE*
CAN ID *00 04 00 14*
Description *Get azimuth brake status*
Typical Interval *5 s*
Data *1 byte (ubyte)*
0x00: brake disengaged
0x01: brake engaged

Name *GET_AZ_ENC*
CAN ID *00 04 00 17*
Description *Position in raw encoder[1] bits at last 20.83 Hz tick*
Typical Interval *48 ms*

[1] Mean value of all reading heads.

Data *4 bytes (uint32): value of azimuth encoder. A uint32 containing the raw encoder value.*

Name *GET_AZ_MOTOR_CURRENTS*
CAN ID *00 04 00 19*
Description *Motor currents in all azimuth axis drive motors*
Typical Interval *5 s*
Data *4 bytes:*
byte 0-1 (int16): value of commanded motor current in centi- amps
byte 2-3 (int16): value of current feedback in centi-amps

Name *GET_AZ_MOTOR_TEMPS*
CAN ID *00 04 00 1A*
Description *Motor temperatures in all azimuth axis drive motors*
Typical Interval *5 s*
Data *4 bytes:*
byte 0 (int8): average value of motor (1^{st} half) [2] temperature in degrees
byte 1 (int8): maximum value of motor(1^{st} half)[3] temperature in degrees
byte 2 (int8): average value of motor (2^{nd} half)[2] temperature in degrees
byte 3 (int8): maximum value of motor(2^{nd} half)[3] temperature in degrees

Name *GET_AZ_MOTOR_TORQUE*
CAN ID *00 04 00 15*
Description *Applied motor torque in all azimuth axis drive motors*
Typical Interval *5 s*
Data *4 bytes:*
byte 0-1 (int16): value of commanded motor torque in daNm
byte 2-3 (int16): value of motor torque feedback in daNm

Name *GET_AZ_SERVO_COEFF_N*
CAN ID *00 04 30 20 – 00 04 30 2F*
Description *Azimuth servo coefficients*
Typical Interval *Rare*
Data *8 bytes. (double)*
Each message contains a different servo loop control parameter as defined by the Contractor's implementation.

[2] Average temperature of 10 semi-sectors in (half AZ motor)
[3] Maximum temperature of anyone of the 10 semi-sectors (half AZ motor).

Name *GET_AZ_STATUS*
CAN ID *00 04 00 1B*
Description *Status of azimuth axis*
Typical Interval *5 s*
Data *8 bytes:*

byte 0 - limit switches (ubyte):
bit0: SW CW prelimit (set = in limit)
bit1: HW CW prelimit (set = in limit)
bit2: HW CW final-limit (set = in limit)
bit3: CW shutdown due to limit condition (set = occurred)
bit4: SW CCW prelimit (set = in limit)
bit5: HW CCW prelimit (set = in limit)
bit6: HW CCW final-limit (set = in limit)
bit7: CCW shutdown due to limit condition (set = occurred)

byte 1 – interlocks(ubyte):
bit0: rotation final limit
bit1: axis HW interlock (set=true)
bit2: override command
bit3: hardstop sense +
bit4: hardstop sense –
bit5: sense +
bit6: sense -

byte 2 - motors (ubyte):
bit0: motor over speed (set = true)
bit1: motors 1st half over current (set = true)
bit2: motors 1st half overheating (set = true)
bit3: motor 2nd half over current (set = true)
bit4: motor 2nd half overheating (set = true)
bit5: drive power on
bit6: DC bus 1
bit7: DC bus 2

byte 3 - motors (ubyte):
bit0: motors power-on/switch failure (set = fault)
bit1: motors enable timeout (set = fault)
bit2: motor 1st half fault (set = fault)
bit3: motor 2n half fault (set = fault)
bit4: motor drivers ready (set=Ready)
bit5: encoder/Hall sensors inconsistency
bit6: following error

byte 4 – encoder (ubyte):

Name	*GET_AZ_STATUS*
CAN ID	*00 04 00 1B*
Description	*Status of azimuth axis*
Typical Interval	*5 s*

bit0: encoder value fault (set = fault)
bit1: absolute encoder position not available (set = true)
bit2: encoder value validation (unset = values ok, set = values old)
bit3: servo oscillation (set = true)
bit4: interpolation board #1 status (set = ok)
bit5: interpolation board #2 status (set = ok)

byte 5 – encoder (ubyte):
bit0: encoder head #1 status (set = fault)
bit1: encoder head #2 status (set = fault)
bit2: encoder head #3 status (set = fault)
bit3: encoder head #4 status (set = fault)
bit4: encoder head #5 status (set = fault)
bit5: encoder head #6 status (set = fault)
bit6: encoder head #7 status (set = fault)
bit7: encoder head #8 status (set = fault)

byte 6 – brakes (ubyte):
bit0: brake position error
bit1: brake wear
bit2: brake local mode
bit3: brake out
bit4: brake disengage timeout
bit5: brake engage timeout

byte 7 - spare

Name	*GET_AZ_ENCODER_OFFSET*
CAN ID	*00 04 00 1C*
Description	*Offset between raw encoder reading and azimuth position excluding contribution from pointing and metrology corrections*
Typical Interval	*Rare*
Data	*4 bytes (int32): An int32 containing the encoder offset.*

Name	*GET_CAN_ERROR*
CAN ID	*00 07 00 01*
Description	*Number of CAN bus errors since power-up and error code of last error*
Typical Interval	*(debug)*

Name *GET_CAN_ERROR*
CAN ID *00 07 00 01*
Description *Number of CAN bus errors since power-up and error code of last error*
Typical Interval *(debug)*
Data *4 Bytes*
bytes 0-1 (uint16)
count of CAN errors since power up.
byte 2 (ubyte)
(reserved)
byte 3 (ubyte)
Error code of last CAN error. Codes are those defined by Intel 82527 CAN Controller as follows:
0x00: No error
0x01: Stuff error
0x02: Form error
0x03: Ack error
0x04: Bit1 error
0x05: Bit 0 error
0x06: CRC error

Name *GET_EL_TRAJ_CMD*
CAN ID *00 04 00 03*
Description *Position in turns and velocity in turns/sec set with the last EL_TRAJ_CMD.*
Typical Interval *Rare*
Data *8 bytes.*
Bytes 0-3: Fixed point number as described in EL_POSN_RSP representing turns.
Bytes 4-7: Fixed point number representing "velocity" in turns/sec.
Returns zero values if no EL_TRAJ_CMD has been given.

Name *GET_EL_BRAKE*
CAN ID *00 04 00 04*
Description *Get elevation brake status*
Typical Interval *5 s*
Data *1 byte (ubyte)*
0x00: brake disengaged
0x01: brake engaged

Name *GET_EL_ENC*
CAN ID *00 04 00 07*

Description	*Position in raw encoder[4] bits at last 20.83 Hz tick*
Typical Interval	*48 ms*
Data	*4 bytes* *bytes 0-3 (uint32)* *value of elevation encoder in raw encoder bits*

Name	*GET_EL_MOTOR_CURRENTS*
CAN ID	*00 04 00 09*
Description	*Actual motor currents in all elevation axis drive motors*
Typical Interval	*5 s*
Data	*4 bytes:* *byte 0-1 (int16): value of commanded motor current in centi- amps* *byte 2-3 (int16): value of current feedback in centi-amps*

Name	*GET_EL_MOTOR_TEMPS*
CAN ID	*00 04 00 0A*
Description	*Motor temperatures in all elevation axis drive motors*
Typical Interval	*5 s*
Data	*4 bytes:* *byte 0 (int8): average value of motor (1st half) [5] temperature in degrees* *byte 1 (int8): maximum value of motor(1st half)[6] temperature in degrees* *byte 2 (int8): average value of motor (2nd half) [7] temperature in degrees* *byte 3 (int8): maximum value of motor(2nd half[8] temperature in degrees*

Name	*GET_EL_MOTOR_TORQUE*
CAN ID	*00 04 00 05*
Description	*Applied motor torque in all elevation axis drive motors*
Typical Interval	*5 s*
Data	*4 bytes:* *byte 0-1 (int16): value of commanded motor torque in daNm* *byte 2-3 (int16): value of motor torque feedback in daNm*

Name	*GET_EL_SERVO_COEFF_N*
CAN ID	*00 04 30 10 – 00 04 30 1F*
Description	*Azimuth servo coefficients*
Typical Interval	*Rare*

[4] Mean value of all reading heads.
[5] Average temperature of 10 semi-sectors in (half EL motor)
[6] Maximum temperature of anyone of the 10 semi-sectors (half EL motor).
[7] Average temperature of 10 semi-sectors in (half EL motor)
[8] Maximum temperature of anyone of the 10 semi-sectors (half EL motor).

Data *8 bytes. (double)*
Each message contains a different servo loop control parameter as defined by the Contractor's implementation.

Name *GET_EL_STATUS*
CAN ID *00 04 00 0B*
Description *Status of elevation axis*
Typical Interval *5 s*
Data *8 bytes:*

byte 0 - limit switches (ubyte):
bit0: SW UP prelimit (set = in limit)
bit1: HW UP prelimit (set = in limit)
bit2: HW UP final-limit (set = in limit)
bit3: UP shutdown due to limit condition (set = occurred)
bit4: SW DOWN prelimit (set = in limit)
bit5: HW DOWN prelimit (set = in limit)
bit6: HW DOWN final-limit (set = in limit)
bit7: DOWN shutdown due to limit condition (set = occurred)

byte 1 – interlocks(ubyte):
bit0: rotation final limit
bit1: axis HW interlock (set=true)
bit2: override command

byte 2 - motors (ubyte):
bit0: motor over speed (set = true)
bit1: motors 1st half over current (set = true)
bit2: motors 1st half overheating (set = true)
bit3: motor 2nd half over current (set = true)
bit4: motor 2nd half overheating (set = true)
bit5: drive power on
bit6: DC bus 1
bit7: DC bus 2

byte 3 - motors (ubyte):
bit0: motors power-on/switch failure (set = fault)
bit1: motors enable timeout (set = fault)
bit2: motor 1st half fault (set = fault)
bit3: motor 2n half fault (set = fault)
bit4: motor drivers ready (set=Ready)
bit5: encoder/Hall sensors inconsistency
bit6: following error

Name *GET_EL_STATUS*
CAN ID *00 04 00 0B*
Description *Status of elevation axis*
Typical Interval *5 s*

byte 4 – encoder (ubyte):
bit0: encoder value fault (set = fault)
bit1: absolute encoder position not available (set = true)
bit2: encoder value validation (unset = values ok, set = values old)
bit3: servo oscillation (set = true)
bit4: interpolation board #1 status (set = ok)

byte 5 – encoder (ubyte):
bit0: encoder head #1 status (set = fault)
bit1: encoder head #2 status (set = fault)
bit2: encoder head #3 status (set = fault)
bit3: encoder head #4 status (set = fault)

byte 6 – brakes (ubyte):
bit0: brake position error
bit1: brake wear
bit2: brake local mode
bit3: brake out
bit4: brake disengage timeout
bit5: brake engage timeout

byte 7 - spare

Name *GET_EL_ENCODER_OFFSET*
CAN ID *00 04 00 0C*
Description *Offset between raw encoder reading and elevation position excluding contribution from pointing and metrology corrections*
Typical Interval *Rare*
Data *4 bytes (int32): An int32 containing the encoder offset.*

Name *GET_SYSTEM_ID*
CAN ID *00 07 00 04*
Description *Get ACU hardware and software identifiers. Currently only a software revision level is supported, but could be expanded to include hardware identifiers in future.*
Typical Interval *Rare*

Data	*3 Bytes* *byte 0 (ubyte): major revision level* *byte 1 (ubyte): minor revision level* *byte 2 (ubyte): patch level* *ie. 0xXX 0xYY 0xZZ is interpreted as VXX.YY.ZZ*

Name	*GET_IDLE_STOW_TIME*
CAN ID	*00 04 00 25*
Description	*Currently set time for ACU to enter survival stow if no communications received on CAN bus or timing pulse has ceased.*
Typical Interval	*Rare*
Data	*2 bytes: (uint16)* *value representing seconds*

Name	*GET_IP_ADDRESS*
CAN ID	*00 04 00 2D*
Description	*ACU IP address*
Typical Interval	*Rare*
Data	*8 Bytes:* *bytes 0 – 3 (uint32): 32 bit IP address organized as byte1.byte2.byte3.byte4* *bytes 4 - 7 (uint32): 32 bit subnet mask organized as byte1.byte2.byte3.byte4*

Name	*GET_IP_GATEWAY*
CAN ID	*00 04 00 38*
Description	*ACU gateway IP address*
Typical Interval	*Rare*
Data	*4 Bytes:* *bytes 0 – 3 (uint32): 32 bit gateway IP address organized as byte1.byte2.byte3.byte4*

Name	*GET_NUM_TRANS*
CAN ID	*00 07 00 02*
Description	*Number of transactions handled by ACU since power up*
Typical Interval	*(debug)*
Data	*4 Bytes: (uint32)* *count of handled transactions*

Name	*GET_SYSTEM_STATUS*
CAN ID	*00 04 00 23*
Description	*State of miscellaneous related systems*

Typical Interval	*5 s*
Data	*7 Bytes* *byte 0 (ubyte):* *bit0: emergency stop (set = applied)* *bit1: ACU interlock (set = applied)* *bit2: base door interlock (set = applied)* *bit3 base 1 interlock (set = applied)* *bit4 base 2 interlock (set = applied)* *bit 5 AZ skirt 1 interlock (set = applied)* *bit6 AZ skirt 2 interlock (set = applied)* *bit7 acces stair 1 interlock (set = applied)* *byte 1 (ubyte):* *bit0 EL left interlock (set = applied)* *bit1 EL right interlock (set = applied)* *bit2 PCU1 RC interlock (set = applied)* *bit3 PCU 2 D1 interlock (set = applied)* *bit4 PCU 3 Base interlock (set = applied)* *bit5 PCU 4 PLC interlock (set = applied)* *bit6 receiver cabin interlock (set = applied)* *bit7 receiver cabin access door interlock (set = applied)* *byte 2 (ubyte):* *bit0 handrail rc platform interlock (set = applied)* *bit1: ACU booting failure (set = failed)* *bit2: survival stow due to missing commands after idle time (set=applied)* *bit3: survival stow due to missing timing pulse after idle time (set=applied)* *bit4: timing pulse missing (set=missing)* *bit5: ACU task failure (set=failure)* *bi6: timing pulse missed (set=missed)*[9] *byte 3 (ubyte):* *bit0 hydraulic unit generic alarm (set = alarm)* *bit1 fire system status (set = fault)* *bit2 over temperature alarm (set = applied)* *bit3 ventilation skirt status (set = fault)* *byte 4(ubyte):* *bit0 antenna local mode* *bit1 antenna remote mode* *bit2 antenna pcu mode* *bit3 PCU platform connected* *bit4: PCU receiver cabin connected*

[9] this bit is latched and needs to be cleared with CLEAR_FAULT_CMD

Name *GET_SYSTEM_STATUS*
CAN ID *00 04 00 23*
Description *State of miscellaneous related systems*
Typical Interval *5 s*

bit5 PCU basement connected
bit6 PCU4 PLC connected

byte 5(ubyte)
bit0 drives power (set = powered)
bit1 drives (set = PLC control)
bit2 hydraulic pump (set = on)
bit3 skirt ventilation (set = on)
bit4 skirt ventilation cmd output

byte 6(ubyte) (shutter status)
bit0: closing ON (set = switch ON)
bit1: opening ON (set = switch ON)
bit2: lock on (set = movement is enabled)
bit3: timeout
bit4: base ladder interlock
bit5: drives lockout status (set = power off)
bit6: missing dummy socket (set = missing)

Name *GET_PT_MODEL_COEFF_N*
CAN ID *00 04 30 40 – 00 04 30 5F*
Description *Pointing model coefficients to be used in autonomous mode. This is a range of consecutive identifiers reserved for getting the current value of a variable number of coefficients.*
Typical Interval *Rare*
Data *8 bytes in each coefficient. Value representing arcseconds. (double)*
1st elem. – IA azimuth encoder zero offset
2nd elem. – CA collimation error of electromagnetic offset
3rd elem. – NPAE non-perpendicularity of mount AZ & EL axes
4th elem. – AN azimuth axis offset (misalignment north-south)
5th elem. – AW azimuth axis offset (misalignment east-west)
6th elem. – IE elevation encoder zero offset
7th elem. – HECE gravitational flexure correction at the horizon
8th ÷ 16th elem. – reserved

Name *GET_SHUTTER*
CAN ID *00 04 00 2E*
Description *Shutter mechanism status*
Typical Interval *5 s*

Data *1 byte: (ubyte)*
bit0: open position (set = open)
bit1: close position (set = close)
bit2: motor shutter ON (set = switch ON)
bit3: local system error ON (set = error)

Name *GET_STOW_PIN*
CAN ID *00 04 00 24*
Description *Stow pins position*
Typical Interval *5 s*
Data *2 bytes*
byte 0 (ubyte):
bit0: AZ stow pin inserted (set = inserted)
bit1: AZ stow pin released (set = released)
byte 1 (ubyte): [10]
bit0: EL stow pin #1 inserted (set = inserted)
bit1: EL stow pin #1 released (set = released)

Name *GET_STOW_PIN_1*
CAN ID *00 04 00 21*
Description *Stow pins position*
Typical Interval *5 s*
Data *4 bytes*
byte 0 (ubyte):
bit0: AZ stow pin inserted (set = inserted)
bit1: AZ stow pin released (set = released)
bit2: AZ stow pin thermal protection
bit3: AZ stow pin over torque
bit4: AZ stow pin deploying
bit5 AZ stow pin releasing
bit6 AZ stow pin timeout

byte 1 (ubyte):
bit0: EL stow pin #1 inserted (set = inserted)
bit1: EL stow pin #1 released (set = released)
bit2: EL stow pin #1 thermal protection
bit3: EL stow pin #1 over torque
bit4: EL stow pin #1 deploying
bit5 EL stow pin #1 releasing
bit6 EL stow pin #1 timeout

---

[10] Since the AEM antenna has two EL stow pins, the logic is the following: If at least one stow pin is inserted the status is inserted. If both stow pins are released the status is released.

Name *GET_STOW_PIN_1*
CAN ID *00 04 00 21*
Description *Stow pins position*
Typical Interval *5 s*

byte 2 (ubyte)
bit0: EL stow pin #2 inserted (set = inserted)
bit1: EL stow pin #2 released (set = released)
bit2: EL stow pin #2 thermal protection
bit3: EL stow pin #2 over torque
bit4: EL stow pin #2 deploying
bit5 EL stow pin #2 releasing
bit6 EL stow pin #2 deploying timeout

byte3
bit0: stow pin auto/man (set = manual)
bit1: AZ stow pin position error
bit2: EL stow pin #1 position error
bit3: EL stow pin #2 position error
bit4: AZ stow pin centered
bit5: EL stow pin centered

Name *GET_SUBREF_ABS_POSN*
CAN ID *00 04 00 26*
Description *Get absolute position of subreflector mechanism*
Typical Interval *5 s*
Data *6 bytes*
bytes 0-1 (int16):
X axis subreflector absolute position in μm (range -32768 to 32767)
bytes 2-3 (int16):
Y axis subreflector absolute position in μm (range -32768 to 32767)
bytes 4-5 (int16):
Z axis subreflector absolute position in μm (range -32768 to 32767)

Name *GET_SUBREF_DELTA_POSN*
CAN ID *00 04 00 27*
Description *Get delta position of subreflector mechanism*
Typical Interval *5 s*

Name *GET_SUBREF_DELTA_POSN*
CAN ID *00 04 00 27*
Description *Get delta position of subreflector mechanism*
Typical Interval *5 s*
Data *6 bytes*

bytes 0-1 (int16):
X axis subreflector delta position in μm (range -32768 to 32767)
bytes 2-3 (int16):
Y axis subreflector delta position in μm (range -32768 to 32767)
bytes 4-5 (int16):
Z axis subreflector delta position in μm (range -32768 to 32767)

Name *GET_SUBREF_LIMITS*
CAN ID *00 04 00 28*
Description *Get subreflector mechanism limit status*
Typical Interval *5 s*
Data *3 bytes*

byte 0 X axis limit status (ubyte):
bit0: upper software position limit (set = exceeded)
bit1: lower software position limit (set = exceeded)
bit2: upper software rotational limit (set = exceeded)
bit3: lower software rotational limit (set = exceeded)

byte 1 Y axis limit status (ubyte):
bit0: upper software position limit (set = exceeded)
bit1: lower software position limit (set = exceeded)
bit2: upper software rotational limit (set = exceeded)
bit3: lower software rotational limit (set = exceeded)

byte 2 Z axis limit status (ubyte):
bit0: upper software position limit (set = exceeded)
bit1: lower software position limit (set = exceeded)
bit2: upper software rotational limit (set = exceeded)
bit3: lower software rotational limit (set = exceeded)
Hardware switch not used.[11]

11 Regarding the hardware limits:
Each leg has it own limit switch electronics, these works only for the leg itself. If one leg runs into the limit, the power will be cut and the brake will be activated. But only the movement into the direction of hardstop is blocked. It is possible to move the leg back to the operation range. That means an INI will work. PTo be reminded that in normal operation the hexapod will never run into the hardware limit. In this case something is wrong and has to be analyzed before restart.

Name	*GET_SUBREF_ROTATION*
CAN ID	*00 04 00 2A*
Description	*Subreflector rotation position*
Typical Interval	*Rare*
Data	*6 Bytes:* *Bytes 0-1: (int16) X tip in 0.0001 degrees* *Bytes 2-3: (int16) Y tilt in 0.0001 degrees* *Bytes 4-5: (int16) Z rotation in 0.0001 degrees (optional)*

Name	*GET_SUBREF_STATUS*
CAN ID	*00 04 00 29*
Description	*Get subreflector mechanism status*
Typical Interval	*5 s*
Data	*4 Bytes:* *byte0 (ubyte):* *bit0: power monitor (set = powered)* *bit1: over run (tape switch)* *bit2: initialized (set = initialized)* *bit3:is initializing (set = initializing procedure is running)* *bit4: servo state (set = servo is on)* *bit5: connection fault* *bit6: SM override* *byte1 motion (ubyte):* *bit0 strut 1 motion status (set = moving)* *bit1 strut 2 motion status (set = moving)* *bit2 strut 3 motion status (set = moving)* *bit3 strut 4 motion status (set = moving)* *bit4 strut 5 motion status (set = moving)* *bit5 strut 6 motion status (set = moving)* *byte2 controller error status (ubyte):* *bit0 strut 1 controller error(set = error)* *bit1 strut 2 controller error(set = error)* *bit2 strut 3 controller error(set = error)* *bit3 strut 4 controller error(set = error)* *bit4 strut 5 controller error(set = error)* *bit5 strut 6 controller error(set = error)* *byte3 actuator over temperature (if above 85 °C)* *bit0: hexapod temperature monitoring (set = ok)* *bit1: actuator #1 over temperature (set = over temperature)* *bit2: actuator #2 over temperature (set = over temperature)* *bit3: actuator #3 over temperature (set = over temperature)*

Name	*GET_SUBREF_STATUS*
CAN ID	*00 04 00 29*
Description	*Get subreflector mechanism status*
Typical Interval	*5 s*
	bit4: actuator #4 over temperature (set = over temperature) *bit5: actuator #5 over temperature (set = over temperature)* *bit6: actuator #6 over temperature (set = over temperature)*

Name	*GET_METR_MODE*
CAN ID	*00 04 00 31*
Description	*Get metrology mode. The bits corresponding to specific devices assume that these devices will be included in the design.*
Typical Interval	*Rare*
Data	*4 bytes:* *byte 0 (ubyte):* *bit0: standard pointing model enabled* *bit1: tiltmeter compensation enabled* *bit2: temperature compensation enabled* *bit3: metrology correction enabled* *bit4: reserved* *bit5: automatic sub-reflector position correction enabled* *bit6: encoder mount displacement sensor correction enabled* *bit7: reserved* *bytes 1-3: spares*

Name	*GET_METR_EQUIP_STATUS*
CAN ID	*00 04 00 32*
Description	*Get metrology status*
Typical Interval	*5 s*
Data	*4 bytes* *Byte 0 (ubyte)* *bit0: metrology power off* *bit1 tiltmeter power off* *bit2: thermal metrology communication ok* *bit3: right tiltmeter communication ok* *bit4: left tiltmeter communication ok* *bit5: thermal metrology out of range* *bit6: right tiltmeter read out of range* *bit7: left tiltmeter read out of range* *Bytes 1-3: spare*

Name	*GET_METR_COEFF_N*
CAN ID	*00 04 00 50 N – 00 04 00 51*
Description	*Metrology model coefficients to be used in autonomous mode*
Typical Interval	*Rare*
Data	*8 bytes in each coefficient. Value representing arcseconds. (double)* *1st elem. – AN0 (Az axis tilt to be subtracted from tiltmeter readout)* *2nd elem. – AW0 (Az axis tilt to be subtracted from tiltmeter readout)*

Name	*GET_METR_TEMPS_N*
CAN ID	*00 04 40 N*
Description	*Metrology system temperatures. There are up to 100 temperature sensors. This group of monitor points returns them in blocks of 4; that is, N is in the range 0 to 25. This assumes that sensors measuring temperatures will be included in the design.*
Typical Interval	*5 s*
Data	*8 bytes (4 int16 values)* *Temperature measured by temp sensors N*4 thru N*4 + 3 (up to 100 sensors distributed across 25 CAN messages); Value is in multiples of 0.01 degree C* *Temperature values shall indicate if sensor is broken or disabled by returning in the monitor point an extreme value, for example the maximum or minimum value.* *Returned data:* *299.90 deg. C: +overflow* *-299.90 deg. C: -overflow* *299.91 deg. C: disconnected* *299.92 deg. C: N/A (no sensor/sensor disabled)*

Name	*GET_METR_TILT_N*
CAN ID	*00 04 50 N*
Description	*Metrology system tiltmeter readouts. There are 2 tilt values, so N is in the range 0 to 1. This assumes that sensors measuring tilt will be included in the design.*
Typical Interval	*100ms*

Data	*8 bytes (4 int16 values)* *byte 0-1: tilt(c)* *byte 2-3: tiltmeter N+1, tilt(s)* *byte 4-5: tiltmeter N+1, tilt(d)* *byte 6-7: tiltmeter N+1, temperature* *tilts: multiples of 0.01 arcsec* *temperatures: multiples of 0.01 degree C*

Name	*GET_METR_DELTAS*
CAN ID	*00 04 00 34*
Description	*Get AZ and EL total delta correction applied by the metrology to the AZ/EL position.*
Typical Interval	*48 ms*
Data	*8 bytes:* *bytes 0-3: azimuth delta correction* *bytes 4-7: elevation delta correction* *Data format: signed, two's complement, fixed point binary number representing angle from –1 turn to +(1–2^{-31}) turn*

Name	*GET_METR_DELTAS_TEMP*
CAN ID	*00 04 00 33*
Description	*Get AZ and EL total delta correction applied by the metrology due to temperature variations to the AZ/EL position.*
Typical Interval	*48 ms*
Data	*8 bytes:* *bytes 0-3: azimuth delta correction* *bytes 4-7: elevation delta correction* *Data format: signed, two's complement, fixed point binary number representing angle from –1 turn to +(1–2^{-31}) turn*

Name	*GET_METR_DELTAPATH*
CAN ID	*00 04 00 53*
Description	*Error in path length*
Typical Interval	*48 ms*
Data	*4 bytes (int32) value in multiples of 1nm, positive value if path length is longer than nominal value.* *Range: -2147.483648 to +2147.483647 mm*

Name	*GET_POWER_STATUS*
CAN ID	*00 04 00 30*

Description *Get power and UPS status*
Typical Interval *5 s*
Data *2 bytes*

byte 0 (ubyte):
bit0: antenna power source(reflects status of manual power switch, not set = power from ALMA, set = power from transporter)
bit1: UPS line failure
bit2 UPS low battery
bit3 UPS alarm
bit4 UPS load on bypass
bit5 UPS load on inverter

byte1 24VDC monitoring (ubyte):
bit0: aux ready (set = ok)
bit1: interface (set = ok)
bit2: azimuth (set = ok)
bit3: elevation (set = ok)
bit4: interlocks (set = ok)

Name *GET_AC_STATUS*
CAN ID *00 04 00 2C*
Description *Get air conditioning subsystem status*
Typical Interval *5 s*
Data

8 Bytes:

byte 0 (ubyte) ATU:
bit0: ATU resistors overload release
bti1: ATU fan overload release
bit2: lack of flow alarm
bit3: air recirculation devices overload release
bit4: resistors safety thermostat
bit5; differential pressure switch
btt6: manual start/stop request
bit7: fan on

byte 1 (ubyte) ATU:
bit0:- thermal probe S47 fault
bit1: thermal probe S48 fault
bit2: setpoint not reached
bit3: overtemperature alarm
bit4: spare
bit5: spare

Name	*GET_AC_STATUS*
CAN ID	*00 04 00 2C*
Description	*Get air conditioning subsystem status*
Typical Interval	*5 s*
Data	

bit6: watchdog
bit7: spare

byte 2 (ubyte) CHILLER:
bit 0: low pressure
bit 1: high pressure
bit 2: lack of flow alarm
bit 3: pump overload release
bit 4: compressor overload release
bit 5: flow probe
bit 6: manual start/stop request
bit 7: pump on

byte 3 (ubyte) CHILLER:
bit 0: delivery probe fault
bit 1: return probe fault
bit 2: pressure sensor fault
bit 3: cpr command
bit 4: inverter command
bit 5: anti freeze
bit 6: watchdog
bit 7: fan fault

byte 4 (ubyte) CHILLER:
bit 0: inverter fault
bit 1: phase sequence fault
bit 2: spare
bit 3: spare
bit 4: spare
bit 5: spare
bit 6: spare
bit 7: spare

byte 5 (ubyte) HVAC:
bit 0: HVAC disabled
bit 1: Chiller connection OK

bit 2: ATU connection OK
bit 3: spare
bit 4: spare

Name *GET_AC_STATUS*
CAN ID *00 04 00 2C*
Description *Get air conditioning subsystem status*
Typical Interval *5 s*
Data

bit 5: spare
bit 6: spare
bit 7: spare

bytes 6-7: (int16) Temperature of chiller sending Value is in multiples of 0.01 degree C

Name *GET_AC_TEMP*
CAN ID *00 04 00 2B*
Description *GET HVAC calibration volume temperature sensors*
Typical Interval *5 s*
Data *4 Bytes:*
Bytes 0-1: (int16) Temperature sensor 1 (Value is in multiples of 0.01 degree C)
Bytes 2-3: (int16)) Temperature sensor 2 (Value is in multiples of 0.01 degree C)

Name *GET_UPS_OUTPUT_VOLTS*
CAN ID *00 04 00 35*
Description *Output voltages by phase*
Typical Interval *5 s*
Data *6 Bytes:*
Bytes 0-1: (int16) Output voltage phase 1 (V)
Bytes 2-3: (int16) Output voltage phase 2 (V)
Bytes 4-5: (int16) Output voltage phase 3 (V)

Name *GET_UPS_OUTPUT_CURRENT*
CAN ID *00 04 00 36*
Description *Output currents by phase*
Typical Interval *5 s*
Data *6 Bytes:*
Bytes 0-1: (int16) Output current phase 1 (A)
Bytes 2-3: (int16) Output current phase 2 (A)
Bytes 4-5: (int16) Output current phase 3 (A)

Name *GET_ANTENNA_TEMPS*
CAN ID *00 04 00 37*
Description *Antenna temperatures*
Typical Interval *5 s*
Data *4 Bytes:*

Name	*GET_ANTENNA_TEMPS*
CAN ID	*00 04 00 37*
Description	*Antenna temperatures*
Typical Interval	*5 s*
	Bytes 0-1: (int16) Air temperature in receiver cabin. Value is in multiples of 0.01 degree C *Bytes 2-3: (int16) Temperature in pedestal. Value is in multiples of 0.01 degree C*[12] *Temperature values shall indicate if sensor is broken or disabled by returning in the monitor point an extreme value, for example the maximum or minimum value.* *Returned data:* *299.90 deg. C: +overflow* *-299.90 deg. C: -overflow* *299.91 deg. C: disconnected* *299.92 deg. C: N/A (no sensor/sensor disabled)*

Name	*GET_SW_REV_LEVEL*
CAN ID	*00 07 00 00*
Description	*Revision level of vendor ACU code*
Typical Interval	*(debug)*
Data	*3 Bytes* *byte 0 (ubyte): major revision level* *byte 1 (ubyte): minor revision level* *byte 2 (ubyte): patch level* *ie. 0xXX 0xYY 0xZZ is interpreted as VXX.YY.ZZ*

Name	*SELFTEST_RSP*
CAN ID	*00 04 00 40*
Description	*Get self-test status*
Typical Interval	*Rare*
Data	*5 Bytes:* *Byte 0: bit 0: self-test running (set = running)* *bit 1: self-test completed (set = completed)* *bit 2: self-test failed (set = failed)* *Bytes 1-2 (int16): number of failing tests* *Bytes 3-4 (int16): number of errors on the self-test error stack*

[12] The temperature of thermal sensor NR1 is returned; this sensor which is located on the base structure. The temperature of the steel structure is measured.

Name	*SELFTEST_ERR*
CAN ID	*00 04 00 41*
Description	*Reads one entry from the self test failure stack*
Typical Interval	*Rare*
Data	*6 Bytes* [13] *Bytes 0-1 (int16): number of failed test* *Bytes 2-5 (float): measured value, if applicable*

Name	*SELFTEST_ERR_1*
CAN ID	*00 04 00 42*
Description	*Reads one entry from the self test failure stack*
Typical Interval	*Rare*
Data	*8 Bytes* [14] *Bytes 0-1 (int16): number of failed test* *Bytes 2-3 (int16): error code* *0 -> test failed no detailed information available* *1 -> test not executed, due to failed previous required test* *2 ->* *Bytes 4-7 (float): measured value, if applicable*

[13] If no error, return length is 0
[14] If no error, return length is 0.

5.1.3.4 Control Points in Detail

Name *ACU_MODE_CMD*[15]
CAN ID *00 04 10 22*
Description *Set current ACU operational and access modes*
Typical Interval *Rare*
Data *1 Byte:*
Byte 0 Axis Mode: ubyte
Bits 0-3: Azimuth Mode
Bits 4-7: Elevation Mode
Axis Mode values:
0x0 SHUTDOWN
0x1 STANDBY
0x2 ENCODER
0x3 AUTONOMOUS
0x4 SURVIVAL STOW
0x5 MAINTENANCE STOW[16]

Name *ACU_TRK_MODE_CMD*[17]
CAN ID *00 04 10 20*
Description *Current tracking mode for ACU.*
Typical Interval *Rare*
Data *1 byte: Axis Tracking Modes: ubyte*
Axis Tracking Mode values:
0x0 CONTINUOUS SIDEREAL TRACKING
0x1 SLEWING (Similar to continuous sidereal tracking, but may allow lower gains. Normally used during preset to a new object)
0x2 FAST SWITCHING
0x3 ON THE FLY TOTAL POWER MAPPING
0x4 ON THE FLY TOTAL INTERFEROMETRIC MOSAICKING

Name *AZ_TRAJ_CMD*
CAN ID *00 04 10 12*

---

[15] While the ACU is transitioning to a particular mode, repeating the ACU_MODE command to send it to the same mode will not lead to any error (i.e. there will be no action in this case).
[16] This command is used with reduced max ACC/VEL on the drives setting for the "survival stow" condition as requested into the technical specification (ice, snow, 30m/s wind, etc.). The relevant reductions are reported into the Drive Systems and ACU Design Reports.
[17] The tracking mode is automatically set to *CONTINUOUS SIDEREAL TRACKING whenever one axis switches from STANDBY to ENCODER or AUTONOMOUS*

Description	*Desired position in turns and velocity in turns/sec at 20.83 Hz tick subsequent to next. The values are treated differently depending on the ACU's operational mode. In AUTONOMOUS mode, the values are corrected by the pointing model and possibly by metrology. In ENCODER mode, no corrections are applied.*
Typical Interval	*48 ms*
Data	*8 bytes.* *Bytes 0-3: Fixed point number as described in AZ_POSN_RSP representing turns.* *Bytes 4-7: Fixed point number representing "velocity" in turns/sec.*

Name	*EL_TRAJ_CMD*
CAN ID	*00 04 10 02*
Description	*Desired position in turns and velocity in turns/sec at 20.83 Hz tick subsequent to next. The values are treated differently depending on the ACU's operational mode. In AUTONOMOUS mode, the values are corrected by the pointing model and possibly by metrology. In ENCODER mode, no corrections are applied*
Typical Interval	*48 ms*
Data	*8 bytes.* *Bytes 0-3: Fixed point number as described in EL_POSN_RSP representing turns.* *Bytes 4-7: Fixed point number representing "velocity" in turns/sec*

Name	*CLEAR_FAULT_CMD*
CAN ID	*00 04 10 21*
Description	*Clear all existing fault condition flags. Where faults are still active, the corresponding bit will still be set for status monitoring points. Where a fault condition has been rectified, the corresponding bit will no longer be set in the status monitor point.*
Typical Interval	*Rare*
Data	*1 byte: 0x01 will be sent to activate the command* *0xFF: defined as NO_ACTION*

Name	*RESET_ACU_CMD*[18]
CAN ID	*00 04 10 2F*
Description	*Perform a soft reboot of the ACU or peripherals*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *bit0: complete ACU and peripherals reboot* *bit1: metrology subsystem reboot* *bit2: subreflector subsystem reboot*

[18] This command is only accepted when both axis are in SHUTDOWN mode.

Name	*RESET_ACU_CMD_1*[19]
CAN ID	*00 04 10 32*
Description	*Perform a soft reboot of the ACU or peripherals*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *bit0: complete ACU and peripherals reboot* *bit1: metrology subsystem reboot* *bit2: subreflector subsystem reboot* *bit3:AZ drives reset*[20] *bit4: EL drives reset*[21] *bit5: subreflector subsystem reboot with tape switch override*[22] *bit 6: TE re-synch command* *bit 7: Encoders-Hall sensors inconsistency, error clear command*[23]

Name	*SET_AZ_BRAKE*
CAN ID	*00 04 10 14*
Description	*Engage or disengage azimuth brake. This command should be rejected if setting the brake will damage the antenna (if for example the antenna is moving).*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *0x00: disengage brake* *0x01: engage brake* *0xFF: defined as NO_ACTION*

Name	*SET_AZ_SERVO_COEFF_N*[24]
CAN ID	*00 04 20 20 – 00 04 20 2F*
Description	*Azimuth servo coefficients. These values should not be persistent and should default to static "safe" values when the ACU is rebooted.*
Typical Interval	*Rare*
Data	*8 bytes. (double)* *Each message contains a different servo loop control parameter as defined by the Contractor's implementation.*

[19] This command is only accepted when both axis are in SHUTDOWN mode
[20] After this command, a CLEAR_FAULT_CMD shall be sent.
[21] Same as note 20.
[22] When the Subreflector mechanism accidentally goes to the final limit tape, no further operation is possible. This command runs a routine to bring the Hexapod again into the center This is a maintenance command before issuing this command a check on what happened must be performed.
[23] Same as note 20.
[24] This command is not a direct value assignment and it takes about 20ms time before the value can be checked with its related monitor point

Name *SET_AZ_SERVO_DEFAULT*
CAN ID *00 04 10 17*
Description *Reset azimuth servo coefficients to default "safe" values*
Typical Interval *Rare*
Data *1 byte: (ubyte)*
0x01: Set servo coefficients to default values.
0xFF is defined as NO_ACTION

Name *INIT_AZ_ENC_ABS_POS*
CAN ID *00 04 10 18*
Description *Starts the automatic routine to determine the azimuth encoder absolute position*
Typical Interval *Rare*
Data *1 byte: (ubyte)*
0x01:Starts automatic routine
0xFF: defined as NO_ACTION

Name *SET_EL_BRAKE*
CAN ID *00 04 10 04*
Description *Engage or disengage elevation brake. This command should be rejected if setting the brake will damage the antenna (if for example the antenna is moving).*
Typical Interval *Rare*
Data *1 byte: (ubyte)*
0x00: disengage brake
0x01: engage brake
0xFF: defined as NO_ACTION

Name *SET_EL_SERVO_COEFF_N*[25]
CAN ID *00 04 20 10 – 00 04 20 1F*
Description *Elevation servo coefficients. These values should not be persistent and should default to static "safe" values when the ACU is rebooted.*
Typical Interval *Rare*
Data *8 bytes. (double)*
Each message contains a different servo loop control parameter as defined by the Contractor's implementation.

25 This command is not a direct value assignment and it takes about 20ms time before the value can be checked with its related monitor point

Name	*SET_EL_SERVO_DEFAULT*
CAN ID	*00 04 10 07*
Description	*Reset elevation servo coefficients to default "safe" values*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *0x01: Set servo coefficients to default values.* *0xFF: defined as NO_ACTION*

Name	*INIT_EL_ENC_ABS_POS*
CAN ID	*00 04 10 08*
Description	*Starts the automatic routine to determine the elevation encoder absolute position*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *0x01:Starts automatic routine* *0xFF: defined as NO_ACTION*

Name	*SET_IDLE_STOW_TIME*
CAN ID	*00 04 10 25*
Description	*Time for ACU enter survival stow if no communications received on CAN bus or timing pulse has ceased.*
Typical Interval	*Rare*
Data	*2 bytes: (uint16)* *Value representing seconds. Setting the value to 0 turns this feature off.*

Name	*SET_IP_ADDRESS*[26]
CAN ID	*00 04 10 24*
Description	*Set the new ACU IP address*
Typical Interval	*Rare*
Data	*8 bytes:* *bytes 0 - 3 (uint32): 32 bit IP address organized as byte1.byte2.byte3.byte4* *bytes 4 - 7 (uint32): 32 bit subnet mask organized as byte1.byte2.byte3.byte4*

Name	*SET_IP_GATEWAY*[27]
CAN ID	*00 04 10 38*
Description	*Set the new ACU gateway IP address*

[26] SET_IP_ADDRESS: see SET_IP_GATEWAY footnote.

[27] SET_IP_GATEWAY: this command, as the SET_IP_ADDRESS, works by changing the network configuration file which is used at boot-time. Therefore, this command should be followed by a reboot, to make it effective. As it takes some time (up to 1 second) to process this command, the ACU should not be rebooted immediately after the SET_IP* command(s).

Typical Interval *Rare*

Data *4 bytes:*
bytes 0 - 3 (uint32): 32 bit gateway IP address organized as byte1.byte2.byte3.byte4

Name *SET_PT_MODEL_COEFF_N*

CAN ID *00 04 20 40 – 00 04 20 5F*

Description *Pointing model coefficients to be used in autonomous mode. This is a range of consecutive identifiers reserved for setting a variable number of coefficients for the antenna. These values should not be persistent and should default to static "safe" values when the ACU is rebooted. This assumes that sensors measuring displacement will be included in the design.*

Typical Interval *Rare*

Data *8 bytes in each coefficient. Value representing arcseconds. (double)*
1st elem. – IA azimuth encoder zero offset
2nd elem. – CA collimation error of electromagnetic offset
3rd elem. – NPAE non-perpendicularity of mount AZ & EL axes
4th elem. – AN azimuth axis offset (misalignment north-south)
5th elem. – AW azimuth axis offset (misalignment east-west)
6th elem. – IE elevation encoder zero offset
7th elem. – HECE gravitational flexure correction at the horizon
8th ÷ 16th elem. – reserved

Name *SET_STOW_PIN*

CAN ID *00 04 10 2D*

Description *Insert or release the azimuth and/or elevation stow pins. The command data could be a combination (by OR operator) of the reported values*

Typical Interval *Rare*

Data *2 bytes*
byte 0 (ubyte):
0x01: insert AZ stow pin
0x02: release AZ stow pin
byte 1 (ubyte):[28]
0x01: insert EL stow pins
0x02: release EL stow pins

Name *SET_SUBREF_ABS_POSN*

CAN ID *00 04 10 29*

Description *Set the new subreflector absolute position. Setting an absolute position shall reset any delta position*

[28] The command acts on both stow pins.

Typical Interval *0.5*
Data *6 bytes*
bytes 0-1 (int16):
new X axis subreflector desired absolute position in μm (range -32768 to 32767)
bytes 2-3 (int16):
new Y axis subreflector desired absolute position in μm (range -32768 to 32767)
bytes 4-5 (int16):
new Z axis subreflector desired absolute position in μm (range -32768 to 32767)

Name *SET_SUBREF_DELTA_POSN*
CAN ID *00 04 10 2A*
Description *Set the new subreflector delta position*
Typical Interval *0.5*
Data *6 bytes*
bytes 0-1 (int16):
new X axis subreflector desired delta position in μm (range -32768 to 32767)
bytes 2-3 (int16):
new Y axis subreflector desired delta position in μm (range -32768 to 32767)
bytes 4-5 (int16):
new Z axis subreflector desired delta position in μm (range -32768 to 32767)

Name *SUBREF_DELTA_ZERO_CMD*
CAN ID *00 04 10 2B*
Description *Reset the subreflector: setting the subreflector to the absolute position and clearing the delta position*
Typical Interval *0.5*
Data *1 bytes (ubyte)*
0x01: perform the command
0xFF is defined as NO_ACTION

Name *SET_SUBREF_ROTATION*[29]
CAN ID *00 04 10 28*
Description *Rotation control of subreflector*
Typical Interval *0.5 s*
Data *6 Bytes:*
Bytes 0-1: (int16) X tip in 0.0001 degrees (range ±1.5deg)

[29] The parameter range-check is performed by the hexapod control task, and it may take up to 2s before an out-of-range error is reported onto the ACU error stack.

Bytes 2-3: (int16) Y tilt in 0.0001 degrees (range ±1.5deg)
Bytes 4-5: (int16) not used

Name	*SET_METR_MODE*
CAN ID	*00 04 10 26*
Description	*Enable or disable the metrology encoder value correction*
Typical Interval	*Rare*
Data	*4 bytes:* *byte 0 (ubyte):* *bit0: standard pointing model enable/disable (1=enable 0=disable)* *bit1: tiltmeter compensation enable/disable* *bit2: temperature compensation enable/disable* *bit3: metrology correction enable/disable* *bit4: spare* *bit5: automatic sub-reflector position correction enable/disable* *bit6: encoder mount displacement sensor correction enable/disable* *bit7: reset wind metrology reference to zero* *bytes 1 – 3: (spare)*

Name	*SET_METR_COEFF_N*
CAN ID	*00 04 10 50 N – 00 04 10 51*
Description	*Metrology model coefficients to be used in autonomous mode. These values should not be persistent and should default to static "safe" values when ACU is rebooted*
Typical Interval	*Rare*
Data	*8 bytes in each coefficient. Value representing arcseconds (double)* *1st elem. – AN0 (Az axis tilt to be subtracted from tiltmeter readout)* *2nd elem. – AW0 (Az axis tilt to be subtracted from tiltmeter readout)*

Name	*SET_SHUTTER*
CAN ID	*00 04 10 2E*
Description	*Set position of computer actuated shutter*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *0x00: close shutter* *0x01: open shutter* *0xFF: defined as NO_ACTION*

Name	*SELFTEST_CMD*
CAN ID	*00 04 10 30*
Description	*Execute self test* *This command is accepted in Shutdown mode only.*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *0x01: ACU (AZ, EL) starts FULL self test. (command accepted only in Shutdown mode)* *0xFF is defined as NO_ACTION*

Name	*SELFTEST_CMD_1*
CAN ID	*00 04 10 31*
Description	*Execute self test* *This command is accepted in Shutdown mode only.*
Typical Interval	*Rare*
Data	*1 byte: (ubyte)* *0x00: abort self test* *0xFE: ACU (AZ, EL) starts FULL self test. (command accepted only in Shutdown mode)* *0x01: ACU properly booted (get)* *0x02: Power* *0x03: AZ Encoder interface* *0x04: AZ Drives interface* *0x05: AZ Brakes* *0x06: AZ Move* *0x07: AZ Encoder Init* *0x08: AZ End Stops* *0x09: AZ Stow Pin* *0x09: Tiltmeters (Metrology)* *0x0A: AZ Axis Attitude* *0x0B: EL Encoder interface* *0x0C: EL Drives interface* *0x0D: EL Brakes* *0x0E: EL Move* *0x0F: EL Encoder Init* *0x10: EL Balancing* *0x11: EL End Stops* *0x12: EL Stow Pins* *0x13: Feed Shutter* *0x14: Subreflector Mechanism* *0x15: AZ Skirt ventilation* *0x16: Smoke Detectors System* *0x17: HVAC system*

Name *SELFTEST_CMD_1*
CAN ID *00 04 10 31*
Description *Execute self test*
This command is accepted in Shutdown mode only.
Typical Interval *Rare*
0x18: Thermal Sensors (Metrology)
0x19: UPS
0xFF: defined as NO_ACTION

Name *SET_AIR_CONDITIONING*
CAN ID *00 04 10 27*
Description *AIR conditioning control*
Typical Interval *Rare*
Data *Bytes 6:*

Byte 1: (ubyte)
bit0: HVAC enable (general command)
bit1: HVAC disable (general command)
bit2: ATU enable
bit3: ATU disable
bit4: Chiller enable
bit5: Chiller disable
bit6: ATU alarm reset
bit7: Chiller alarm reset

byte 2: (ubyte)
bit0: change ATU temperature set point
bit1 change Chiller temperature set poing

bytes 3-4: (int16)
Air temperature of receiver cabin. Value is in multiples of 0.01 degree C (the corresponding bit in byte0 must be set)

bytes 5-6: (int16)
Temperature of chiller (water-glycol mixture) Value is in multiples of 0.01 degree C (the corresponding bit in byte 2 must be set)

Name *DUMP_ERROR_LOG*
CAN ID *00 04 10 70*
Description *Error log dump and buffer control*
Typical Interval *Rare*

Data *1 byte: (ubyte)*
0x01: Dump error log (dump will be done to /var/log/ACU/error.log)
0x02: Diagnostic buffer dump (dump will be done to /var/log/ACU/bufX[30])
0x03: Diagnostic buffer setup[31]
0xFF is defined as NO_ACTION

[30] */var/log/ACU/bufX: here X stays for a number going from 0 to 15. 16 is the number of available buffers. At each dump command the previous dumped data will be overwritten.*
[31] For detail of configuration file refer to the Software Maintenance Manual

5.1.3.5 ACU Position Request Timing

The monitor requests AZ_POSN_RESP and EL_POSN_RESP are specified to return the position at the last 20.83Hz pulse (here illustrated as TE) and 24 ms before. The timing of monitor requests is specified in [RD01]. Figure 2 below illustrates the timing.

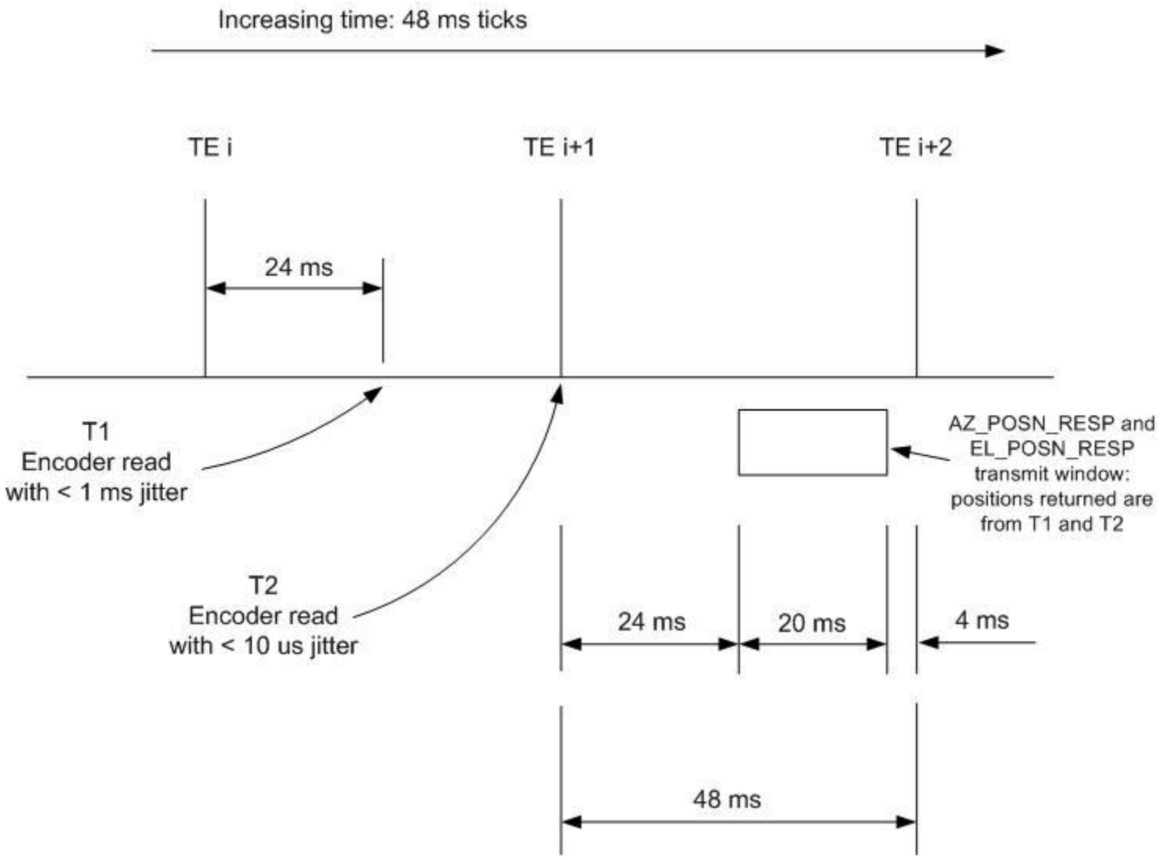

Figure 2: ACU Position Request Timing

5.1.3.6 ACU Trajectory Command Timing

The control commands AZ_TRAJ_CMD and EL_TRAJ_CMD are specified to send the desired position and velocity to the ACU for the 20.83 tick subsequent to the next. The 20.83 ticks are here illustrated as TE. The timing of control commands are specified in [RD01]. Figure 3 below illustrates the timing.

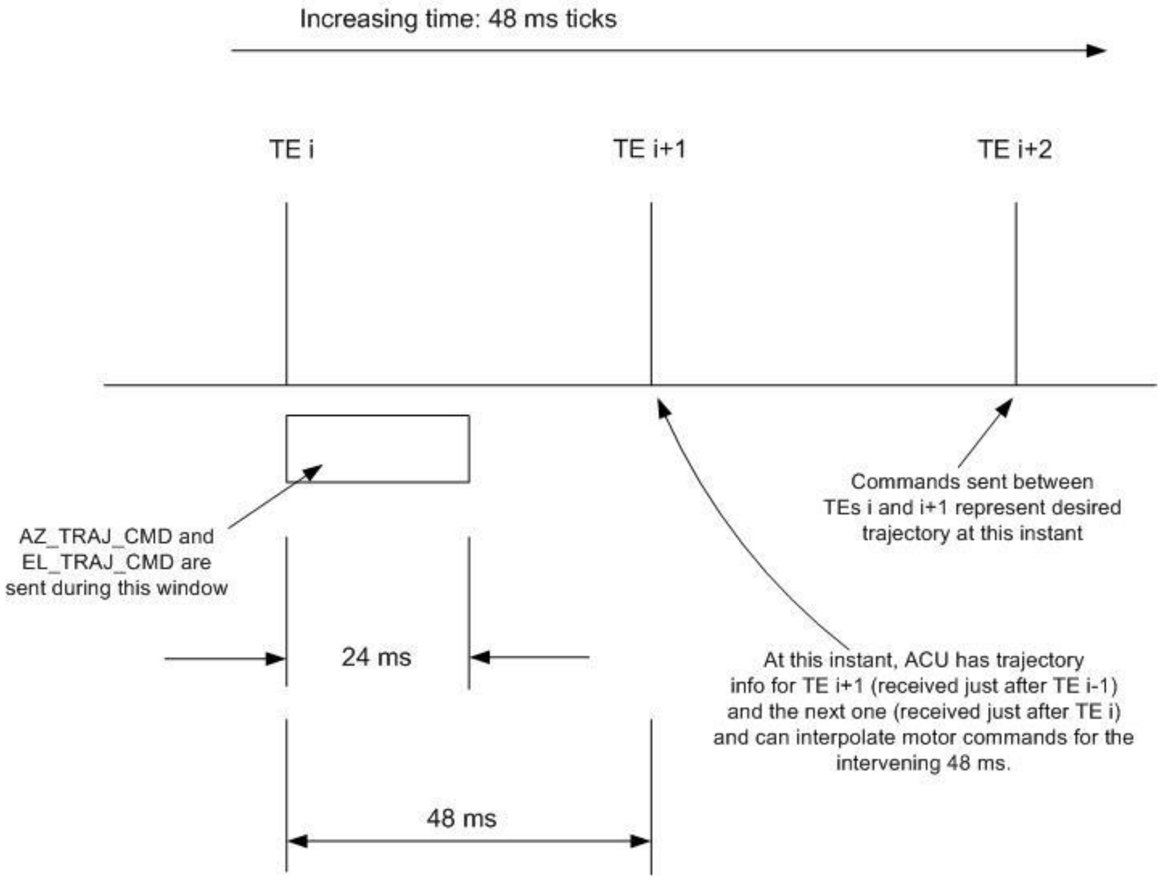

Figure 3: ACU Trajectory Command Timing

If the _TRAJ_CMD (#1) at for example TE i arrives to the ACU after the 24 ms command window, the ACU shall still try to apply it so that the antenna can reach the desired position at TE i+2. In this case, the ACU shall notify the ABM by putting a “trajectory command delayed error” onto the error stack.
If it is too late to apply the command for TE i+2, the ACU may schedule it for one TE later (TE i+3). If the ACU receives another trajectory command (#2) just after TE i+1valid for TE i+3, then it shall discard the one which arrived too late to be valid for TE i+2 (#1) and apply the correct one (#2).

If the ACU receives more than one trajectory command in one TE window, only the first one is valid and all other commands shall be discarded. The ACU shall put a “trajectory command duplicate error” onto the error stack for each discarded command.

5.1.3.7 ACU Trajectory Commands

The trajectory commands sent to the ACU (AZ_TRAJ_CMD or EL_TRAJ_CMD) contains position and velocity for the antenna to reach on the second time event (TE) after the command was sent to the ACU. If the velocity given is not consistent with the position then

the antenna shall try to reach the commanded position as soon as possible respecting the velocity and acceleration limits.

5.1.3.8 ACU Trajectory Command during OTP Mapping and Interferometric Mapping

In order to facilitate for the antenna to follow the path of the OTP Mapping and Interferometric Mapping the ABM will generate a special path for the ACU to indicate the coming turnaround. After the time t_k the ACU trajectory commands will leave the commanded track and jump to the extrapolated datapoints (intermediate datapoints, violet path) from the track after the turnaround, see figure 4 below. The following conditions apply :

- Trajectory commands sent to the ACU (AZ_TRAJ_CMD or EL_TRAJ_CMD) contains position and velocity of the target
- The ACU is set to tracking mode (ENCODER or AUTONOMOUS) and subtracking mode ON THE FLY TOTAL POWER MAPPING or ON THE FLY TOTAL INTERFEROMETRIC MAPPING

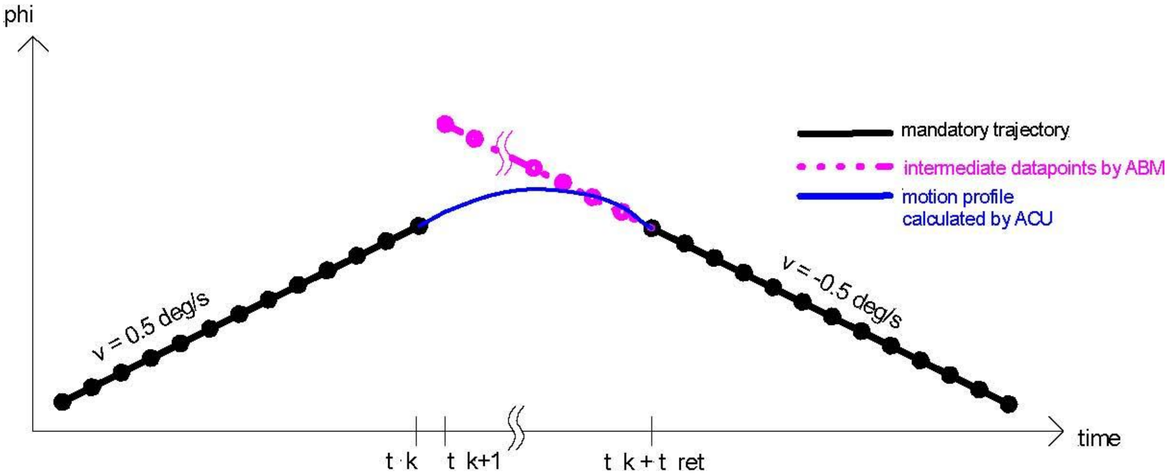

Fig 4: ABM trajectory commands during Turnaround

The time t_k in figure 4 is 9 timing events (TE) or appr. 0.4 seconds before the turnaround and t_k+t_{ret} is 9 TEs after the turnaround. The trajectory at t_{k+1} is on the extrapolated path (violet path) after the turnaround although the turnaround has not occurred yet. This is to signal to the ACU that the turnaround will come. The antenna shall not follow the interpolated path but try to follow the intended path (motion profile, blue path) as close as possible, see motion profile calculated by ACU in figure 4. The commanded velocity of the trajectory at t_{k+1} is equal to –velocity at t_k.

5.1.3.9 Other Signals

The contractor shall provide details on the procedure for setting encoder zeros and any other encoder calibration required. Monitor and control messages required to support such procedures shall be negotiated between the Contractor and ALMA during the design phase.

The contractor shall provide the algorithm for calculating from the raw encoder value the fully corrected antenna position given by GET_AZ/EL_POSN_RSP. If there are other parameters than the encoder offset needed for these calculations then monitor points to make them accessible shall be added.

For actions not terminating instantaneously there shall be an indication in a monitor point when it has terminated.

5.1.4 ACU Modes of Operation

At any time, the ACU may be in one and only one of the following operating modes:

Shutdown: brakes set, no power to motors
Standby: ready to drive, brakes set
Velocity: rate loop driving of axes from local handset
Encoder: drive so encoders equal commanded position
Autonomous: drive so boresight equals commanded position. That is, the commanded position is corrected by the pointing model and any activated metrology.[32]
Survival Stow: drive to survival stow position
Maintenance Stow: drive to maintenance stow position

Simultaneously, the ACU may be in either of two access modes, Local or Remote. When remote access is selected, the controller responds to a set of commands via the CAN bus as defined in Section 5.1.3 above. When Local access is selected, commands received from the digital interface are ignored (but monitor requests are still accepted and processed) and the antenna may be controlled using the local control panel. Switching between Local and Remote access may be done only from the local control panel.

Upon changing to Local or Remote access mode and at power-up and reset, the controller automatically enters the Shutdown operating mode in both axes. Not all operating modes may be entered from either access mode; see Table 8. In addition, Survival Stow mode or Shutdown may be entered automatically when the ACU detects certain fault conditions, regardless of the selected access mode. This is reflected in the Auto column of Table 8.

[32] Metrology correction needs to be requested explicitly it is not automatic.

Table 8: Modes of Operation

	Mode Allowed?		
Operating Mode	***Local***	***Remote***	***Auto***
Shutdown	*Yes*	*Yes*	*Yes*
Standby	*Yes*	*Yes*	*No*
Active Modes			
Velocity	*Yes*	*No*	*No*
Encoder	*Yes*	*Yes*	*No*
Autonomous	*No*	*Yes*	*No*
Survival Stow	*Yes*	*Yes*	*Yes*
Maintenance Stow	*Yes*	*Yes*	*No*

The following rules govern changes of mode:

- From Shutdown mode, the only change permitted is to Standby mode, and then only if no fault conditions exist
- An active mode (Velocity, Encoder, Autonomous, Survival Stow, Maintenance Stow) may only be entered from Standby mode
- From Survival Stow mode, Shutdown mode is automatically entered upon reaching the stow position and the stow pins are inserted.
- From Maintenance Stow mode, Shutdown mode is automatically entered upon reaching the stow position and the stow pins are inserted.
- When Standby is entered, stow pins shall be automatically released. No additional stow-pin removal commands are required to enter Standby.

Any of the “SET_” control messages defined in Section 5.1.3 shall be accepted in any of the operating modes, provided the ACU is in remote mode, unless there are safety issues involved. All monitor messages (Section 5.1.3) shall be handled regardless of the current operating or access mode. The following table shows which “_CMD” control messages shall be applicable in which operating modes (Remote access mode only).

Table 9: Commands allowed by Operating Mode

Axis Operating Mode	*Commands Accepted*
Shutdown	*ACU_MODE_CMD* *RESET_ACU_CMD*
Standby	*ACU_MODE_CMD*
Velocity	*None*
Encoder	*ACU_MODE_CMD* *AZ_TRAJ_CMD* *EL_TRAJ_CMD*
Autonomous	*ACU_MODE_CMD* *AZ_TRAJ_CMD* *EL_TRAJ_CMD*
Survival Stow	*ACU_MODE_CMD* ~~*RESET_ACU_CMD*~~[33]
Maintenance Stow	*ACU_MODE_CMD* ~~*RESET_ACU_CMD*~~[34]

5.1.4.1 Tracking Sub-mode

The tracking sub-mode is only changed by a new command ACU_TRK_MODE, a new ACU mode set with ACU_MODE_CMD, reset or reboot of the ACU. Default value is 0 = CONTINUOUS SIDEREAL TRACKING.

The tracking mode is only valid in ENCODER or AUTONOMOUS mode.

If the ACU_TRK_MODE command is sent when not in AUTONOMOUS or ENCODER mode an error shall be added to the error stack and the ACU shall ignore the command.

The antenna should be able to transition from one sub-mode at any time.

If the ABM sends trajectories which are not compatible with the defined sub-mode, an error should be raised but the ACU should nevertheless meet the commanded trajectory with degraded performance.

The ACU shall directly follow the trajectory commands sent in all sub-modes.

[33] This command cannot be executed during motion because it use the encoder mode, therefore it shall be eliminated from this table.

[34] This command cannot be executed during motion because it use the encoder mode, therefore it shall be eliminated from this table.

5.2 Reset Signal

The ACU shall initiate a reset of the CAN interface circuitry and reboot the ACU when a 1 ms differential RS485 logical "1" pulse is detected on the CAN bus reset pins (pins 1 and 6 as defined in [RD01]). Note that the "RSTA" and "RSTB" signals shown in Figure 1 denote the RS422A and B lines defined in [RD08].
Note that this reset pulse is not a CAN standard.

5.3 Ethernet Interface

In addition to the monitor and control interface provided by the CAN bus, ALMA requires that the ACU also provide an Ethernet interface to facilitate debugging and testing of the vendor equipment.

In particular, is shall be possible via the Ethernet interface to

- download new versions of the software. Installing a new software version shall only be possible when any of the antenna interlocks (emergency stop) pushbuttons is active.
- for debug, configure and profile the ACU software
- issue a reset command to soft boot the ACU. The reset shall only be possible when the antenna is in shutdown mode.

5.4 Static Parameters

The contractor shall in general face the problem of replacement of units (like the ACU, motors etc.) and shall propose a solution for downloading the correct set of parameters relevant to the replaced unit.

Other static parameters to which ALMA requires access:

- Antenna hardware specific parameters (dependent on the Contractors specific implementation)
- Control loop parameters (generally those corresponding to the rare control points in Section 5.1.3.2. above, which are normally fixed at commissioning, but which may require alteration as components such as motors and encoders are replaced).
- XXX software version should be compiled
- Software parameters such as software version numbers, ACU serial number, antenna number and the CAN node number.

ALMA requires the Contractor to provide access to these parameters and procedures for changing them remotely. It is permissible for such alterations to be made over the CAN bus or Ethernet interface. ~~The exact list of static parameters and the methods for altering them are TBD during the design phase~~.
Details are reported in Software Maintenance Manual ANTD-3335030-3-027-MAN.

5.5 Non standard CAN behaviour

The following behavior is required of the ACU but is not standard to the CAN specification:

- Pin 1 and 6 of the CAN connector are used for a remote reset pulse. The presence of a differential RS485 logical "1" on these pins for 1 millisecond should cause the ACU to reset the CAN bus interface circuitry and reboot the ACU.
- The bus will in a master/slave fashion under the control of ALMA's bus master. The Contractor's ACU shall not initiate transmissions on the CAN bus unless polled by ALMA's bus master.

5.6 Self Test

~~The ACU performs self-test at the time of power-up~~ The ACU performs a status check at the time of power-up but does not perform any active test.
RESET_ACU_CMD does not start self test.
When it receives SELFTEST_CMD it performs a more detailed self-test.
~~While self test is running the ACU does not respond to any commands except SELFTEST_RSP~~. If self test failed the ACU notifies the ABM using SELFTEST_RSP and pushes any error into the self-test error stack.
Selftest shall only be accepted in shutdown mode.
The self-test over the CAN interface shall check the status of all devices and sensors. During execution of the self-test other operating modes may be activated as appropriate, but the mode reported by ACU_MODE_RSP shall be SELFTEST during the entire operation of this test whether drives are activated or not.
After completion of the self-test the ACU shall return to SHUTDOWN mode.

The selftest over the CAN interface can only return limited amount of information due to limited message length (8 bytes). Therefore a more detailed selftest log shall be available over the Ethernet interface.[35] ~~Details TBD with the contractor~~.

5.6.1 Self Test Details

All the details regarding the actions performed by this command are reported into the ACU Design Report (ANTD-333503-3-001-REP).

[35] Note that the test type performed via CAN is the same as the one performer locally, just the logging is less detailed. Additionally it shall be noticed that during self-test execution the ACU responds to all commands as usually, of course safety aspects are checked. But it shall be noticed that improper commands could bring to a failure of a self test.

5.7 Circular Buffer

The ACU shall store the positions readings in a circular buffer sufficient to hold data from for at least 10 seconds. The position data shall be at least the position sampled at each timing event (TE) and at the midpoint between 2 TEs.
The circular buffer shall be stored in a local file on RAM disk. It shall be possible to copy the RAM disk file to an ALMA workstation via ftp or ssh. ~~Format of the file TBD with the contractor~~.

5.7.1 Circular Buffer Details

All the details regarding this diagnostic utility are reported into the ACU Design Report (ANTD-333503-3-001-REP).

6 Safety Issues

Sufficient safety features and interlocks shall be provided by the contractor such that no malfunctioning ALMA interface or software can cause equipment damage or endanger human safety.

ALMA software shall monitor and report situations which are approaching or reaching safety limits. In normal operation, ALMA software shall attempt to prevent the activation of hardware limits to provide a second level of safety margins and to reduce the possibility of reaching such hardware limits. Automatic hardware fail-safe mechanisms shall be applied when limits have been reached and ALMA software shall be able to monitor these.

Actions not performed instantaneously shall be monitored by a timeout and timeout errors shall be reported. Example of such actions are mode switching, insertion of stow pins, moving to stow position. All variables and parameters which are safety relevant shall be monitored. The contractor shall identify all safety relevant parameters.

All the safety relevant parameters are not directly accessible by the operator because they have been inserted into configuration files in order to reduce the risk of erroneous parameter setting. Details of such configuration files, containing the mentioned parameters, are reported into the Software Maintenance Manual ANTD-3335030-3-027-MAN.

The ACU shall monitor and display all of the following conditions and should enter the Shutdown operational mode if any of these conditions are detected:

- Excessive motor current
- Motor overheating
- Servo oscillation
- Limit switch actuation
- Critical sensor faults (especially encoders) or power failure

- Overspeed of azimuth or elevation axis
- Any other safety relevant condition

Loss of the timing signal defined in section 4.4 shall not cause any unexpected movement of the antenna. Instead it shall switch to its internal clock and accept the trajectory commands until the IDLE STOW TIME is reached. Lost timing signal shall be indicated in the antenna status.
Loss of the trajectory command shall not cause any unexpected movement of the antenna. Instead it shall interpolate the position continuing with the last commanded velocity until the IDLE STOW TIME is reached. ~~Lost trajectory command shall be indicated in the az/el status.~~

The Contractor shall analyze all safety relevant situations and propose a strategy for a traceable shut-down in severe situations, like the ones indicated above. In other cases it might be appropriate to implement a retry-policy (to make the system robust) where such a retry is possible and safety-critical aspects are not directly involved.

When the elevation angle is getting above 88.9 degrees the ACU shall automatically close the feed shutter.

End of Document